CURSO PRÁTICO 55 DE PROGRAMAÇÃO DE COMPUTADORES

BASIC

PROGRAMAÇÃO BASIC – PROGRAMAÇÃO DE JOGOS – CÓDIGO DE MÁQUINA

NOVA CULTURAL

Cz$ 39,00

INPUT

Vol. 4 N.º 55

NESTE NÚMERO

PROGRAMAÇÃO BASIC

AFINAL, QUAL É O SEU SOM?

Conversão da onda sonora em informação digital. Armazenagem e reprodução de sons digitais. Programas para o Spectrum e o TRS-Color... 1081

PROGRAMAÇÃO DE JOGOS

INTELIGÊNCIA MILITAR

Transforme seu micro em um adversário habilidoso. Inteligência x força bruta. O uso da heurística. Estratégias sutis..... 1086

PROGRAMAÇÃO BASIC

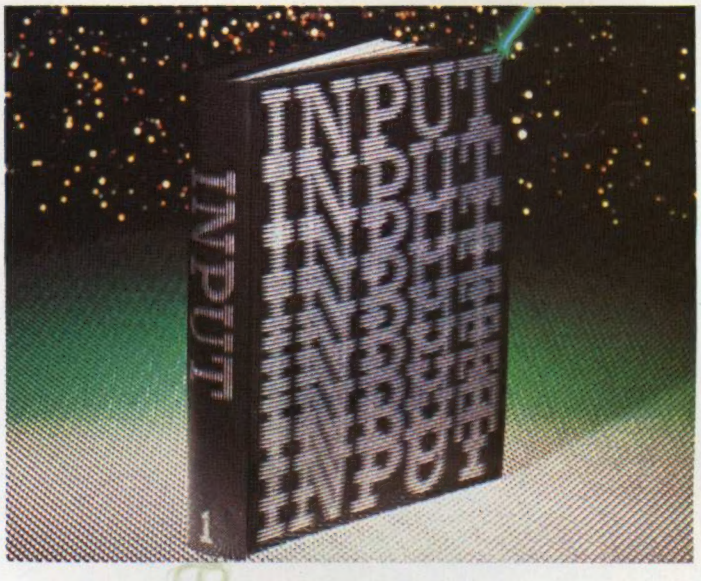
NOVAS MENSAGENS SECRETAS

Como decifrar códigos. Código multiplicativo. Criação de um livro-código. Dicionário de codificação. Programe seu livro-código..... 1091

PROGRAMAÇÃO BASIC

PÁGINAS GRÁFICAS

A persistência da visão. Desenho animado. Páginas gráficas. Movimentação de um cubo. 1096



PLANO DA OBRA

"INPUT" é uma obra editada em fascículos semanais, e cada conjunto de 15 fascículos compõe um volume. A capa para encadernação de cada volume estará à venda oportunamente.

COMPLETE SUA COLEÇÃO

Exemplares atrasados, até seis meses após o encerramento da coleção, poderão ser comprados, a preços atualizados, da seguinte forma: 1. PESSOALMENTE — Por meio de seu jornaleiro ou dirigindo-se ao distribuidor local, cujo endereço poderá ser facilmente conseguido junto a qualquer jornaleiro de sua cidade. Em **São Paulo**, os endereços são: rua Brigadeiro Tobias, 773, Centro; avenida Industrial, 117, Santo André; e no **Rio de Janeiro**: avenida Mem de Sá, 191/193, Centro. 2. POR CARTA — Poderão ser solicitados exemplares atrasados também por carta, que deve ser enviada para DINAP — Distribuidora Nacional de Publicações — Números Atrasados — Estrada Velha de Osasco, 132, Jardim Teresa — CEP 06000 — Osasco — SP. Não envie pagamento antecipado. O atendimento será feito pelo reembolso postal e o pagamento, incluindo as despesas postais, deverá ser efetuado ao se retirar a encomenda na agência do Correio. 3. POR TELEX — Utilize o n.º (011) 33 670 DNAP.
Em **Portugal**, os pedidos devem ser feitos à Distribuidora Jardim de Publicações, Lda. — Qta. Pau Varais, Azinhaga de Fetais — 2 685, Camarate — Lisboa; Apartado 57 — Telex 43 069 JARLIS P.
Atenção: Após seis meses do encerramento da coleção, os pedidos serão atendidos dependendo da disponibilidade do estoque.
Obs.: Quando pedir livros, mencione sempre título e/ou autor da obra, além do número da edição.

COLABORE CONOSCO

Encaminhe seus comentários, críticas, sugestões ou reclamações ao Serviço de Atendimento ao Leitor — Caixa Postal 9442, São Paulo — SP.

NOVA CULTURAL

Editor
VICTOR CIVITA

REDAÇÃO
Diretor Editorial: Carmo Chagas

Editores Executivos: Antonio José Filho, Berta Sztark Amar

Editor Chefe: Paulo de Almeida
Editora de Texto: Ana Lúcia B. de Lucena
Chefe de Arte: Carlos Luiz Batista
Assistentes de Arte: Dagmar Bastos Sampaio, Grace Alonso Arruda, Monica Lenardon Corradi
Secretária de Redação/ Coordenadora: Stefania Crema
Secretários de Redação: Beatriz Hagström, José Benedito de Oliveira Damião, Maria de Lourdes Carvalho, Marisa Soares de Andrade, Mauro de Queiroz

COLABORADORES

Consultor Editorial Responsável: Dr. Renato M. E. Sabbatini (Diretor do Núcleo de Informática Biomédica da Universidade Estadual de Campinas)
Execução Editorial: DATAQUEST Assessoria em Informática Ltda., Campinas, SP
Tradução, adaptação, programação e redação: Abílio Pedro Neto, Aluísio J. Dornellas de Barros, Marcelo R. Pires Therezo, Marcos Huascar Velasco, Raul Neder Porrelli, Ricardo J. P. de Aquino Pereira
Coordenação Geral: Rejane Felizatti Sabbatini

COMERCIAL
Diretor Comercial: Roberto Martins Silveira
Gerente Comercial: Flávio Maculan
Gerente de Circulação: Denise Maria Mozol

PRODUÇÃO
Gerente de Produção: João Stungis
Coordenador de Impressão: Atílio Roberto Bonon

Preparador de Texto/Coordenador: Eliel Silveira Cunha
Preparadores de Texto: Alzira Moreira Braz, Ana Maria Dilguerian, Levon Yacubian, Luciano Tasca, Maria Teresa Galluzzi, Maria Teresa Martins Lopes, Paulo Felipe Mendrone
Revisor/Coordenador: José Maria de Assis
Revisoras: Conceição Aparecida Gabriel, Isabel Leite de Camargo, Ligia Aparecida Ricetto, Maria de Fátima Cardoso, Nair Lucia de Britto
Paste-up: Anastase Potaris, Balduino F. Leite, Edson Donato

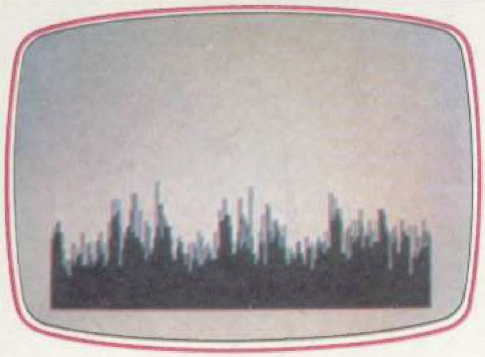
© Marshall Cavendish Limited 1984/85.
© Editora Nova Cultural Ltda., São Paulo, Brasil, 1986.
Edição organizada pela Editora Nova Cultural Ltda.
Av. Brigadeiro Faria Lima, n.º 2000 - 3.º andar
CEP 01452 - São Paulo - SP - Brasil
(Artigo 15 da Lei 5 988, de 14/12/1973).
Esta obra foi composta na AM Produções Gráficas Ltda.
e impressa na Divisão Gráfica da Editora Abril S.A.

AFINAL, QUAL É O SEU SOM?

- ANALÓGICO E DIGITAL
- COMO CAPTAR O SINAL
- O TRAÇADO DO SOM
- ARMAZENAGEM E REPRODUÇÃO DE SONS DIGITAIS

Use o seu microcomputador para explorar a tecnologia da gravação sonora digital. Um simples programa permite a análise do som, assim como a reprodução de um breve trecho musical.

Todo som tem dois componentes: volume e freqüência. Habitualmente, o ouvido humano é capaz de interpretar sinais complexos, transformando as vi-



O traçado imita a onda sonora original.

brações do ar em sons reconhecíveis. Nessa forma, o som é um sinal analógico — isto é, varia continuamente dentro de um intervalo, e qualquer modificação é significativa. Os computadores, ao contrário dos seres humanos, não conseguem reconhecer esse tipo de variação e precisam de um sinal digital. Assim, cada variação é representada por um valor diferente, 0 ou 1, presença ou ausência de sinal.

Mesmo que o computador não seja capaz de interpretar diretamente um sinal sonoro, não é difícil converter a onda sonora analógica de uma melodia, por exemplo, em informação digital.

Essa técnica é a última novidade nos estúdios de gravação, que estão trocando as fitas tradicionais (onde se grava o sinal analógico) por sistemas computadorizados que gravam as músicas em discos (discos de computador). A vantagem de se gravar o som na forma digital baseia-se na facilidade com que se pode combinar o sinal gravado com outros sons. Além disso, uma vez gravado nessa forma, o risco de distorção do sinal devido a limitações do equipamento é bem menor.

MÚSICA EM SEU MICRO

Embora a tecnologia necessária para esse tipo de gravação se restrinja a estúdios com equipamentos altamente sofisticados, a maioria dos microcomputadores tem os recursos básicos que permitem a exploração dessa técnica. Cada vez que carregamos um programa gravado em fita cassete, estamos reproduzindo um sinal que foi registrado na forma digital. Mas quem já ouviu uma fita de programas sabe que ela produz som, ainda que não muito agradável.

Sons e melodias podem ser colocados em seu micro, desde que você recorra às técnicas de programação adequadas (em linguagem de máquina, inclusive). Para isso, é necessário apenas introduzir um sinal analógico através do plugue de gravação e ensinar o computador a interpretá-lo, transformando-o em um sinal digital.

Uma vez feito isto, o programa poderá armazenar o sinal digital na memória, ou mesmo mostrá-lo em forma gráfica, como faz o programa deste artigo. Isso significa que o seu computador será capaz agora de transformar qualquer som em números, que podem ser usados para produzir um gráfico na tela ou ser armazenados na memória para reprodução posterior.

O TRAÇADO DO SOM

O programa pode reproduzir um traçado gráfico que corresponde ao som tocado pelo gravador. Quando este é ligado, uma série de linhas regularmente espaçadas surge na tela (quanto maior a freqüência, mais alta é a linha). No momento em que a tela fica cheia, o traçado desaparece, recomeçando do canto esquerdo do vídeo.

GRAVE O SOM

O programa também permite, numa segunda opção, a gravação digital do sinal de entrada — o tamanho do trecho que se pode gravar é bem limitado, por razões que explicaremos mais adiante. Uma terceira opção possibilita a reprodução do som obtido.

COMO FUNCIONA

Como o computador não pode interpretar diretamente o sinal sonoro analógico, nós o programamos de modo a atribuir valores digitais a esse sinal. O programa verifica repetidamente os sinais na porta de entrada do cassete em intervalos muito curtos de tempo (milhares de vezes por segundo). Esses sinais só podem ser 0 ou 1 — não há valores intermediários como no sinal analógico. A alta velocidade com que a porta é lida repetidas vezes faz com que as variações do sinal digital imitem o sinal analógico.

Imaginemos, por exemplo, que entramos com um sinal de freqüência de 256 Hz (nota C). Esse sinal atinge o pico 256 vezes por segundo, e cada pico dura 1/512 de segundo. Se fizermos a leitura da porta 2000 vezes por segundo, obteremos um pico a cada quatro leituras. É assim que a variação do sinal digital imita a onda sonora analógica — quanto mais rápidas forem as leituras, mais precisa será a conversão analógico-digital.

O programa utiliza os valores digitais obtidos da fita dos modos descritos. Para mostrar a onda sonora graficamente, ele calcula quantos picos — ou quantos "uns" — obteve por unidade de tempo, o que equivale à freqüência média daquele intervalo. Novamente, quanto mais rápida for a leitura, mais preciso será o gráfico.

O processo de gravação é semelhante. O programa toma oito leituras e as armazena em um byte de memória. Esse procedimento é repetido para cada oito leituras. Como um elevado número de leituras é feito em um curto espaço de tempo, o programa utiliza grande quantidade de memória — no Spectrum, por exemplo, toda a memória seria consumida por um trecho musical de apenas oito segundos de duração.

A reprodução do sinal digital constitui o processo inverso, no qual a informação armazenada na memória é enviada ao alto-falante para imitar as vibrações lidas na porta do cassete durante a gravação. Devido às limitações do equipamento, o som produzido está longe de ser de alta-fidelidade.

S

O programa do Spectrum é dividido em duas partes — um programa BASIC com as rotinas de gravação, execução e de traçado gráfico, e uma rotina em linguagem de máquina, para ler a entrada do gravador cassete.

Digite a primeira parte e grave-a utilizando a instrução:

```
SAVE 'ANALYSER' LINE 5
```

```
 10 CLEAR 26000: RESTORE : LET
 t=0: LET x=65368
 20 FOR n=1 TO 108
 30 READ a: POKE x,a
 40 LET t=t+a
 50 LET x=x+1
 60 NEXT n
 70 IF t=12721 THEN  PRINT "OK
 .": STOP
 80 PRINT "ERRO NAS LINHAS 'DA
 TA'": STOP
 90 DATA 14,64,175,8,17,208,7,
 219,254,230,64,185,40,7,62,64
 ,169,79,8,60,8,29,32,239,175,
 178,40,5,30,255,21,24,230,8,
 203,63,6,0,79,201
100 DATA 243,33,144,101,17,80,
255,6,7,219,254,203,119,32,2,
203,254,203,62,16,244,35,125,
187,32,237,124,186,32,233,251,
201,243,33,144,101,17,80,255,6
,8,203,70,40,4
110 DATA 62,0,211,254,62,255,
211,254,203,14,16,240,35,125,
```

```
187,32,233,124,186,32,229,251,
201
```

A seguir, digite a segunda parte, que contém a rotina em código dentro de suas linhas **DATA**. A linha 80 verifica se houve algum erro de digitação nas linhas **DATA**, por meio da soma dos números. Execute (**RUN**) o segundo programa e grave a rotina em código na fita, numa posição imediatamente posterior ao primeiro programa com:

```
SAVE 'ANALYSER' CODE 65368,109
```

Ao ser rodado, o primeiro programa se auto-executará a partir da linha 5, carregando então a rotina em código. Completado o processo, surgirá na tela um menu com as três opções.

A primeira delas solicitará que conectemos o gravador e toquemos algum som. Quando fazemos isso e pressionamos qualquer tecla, é desenhado um gráfico contínuo da evolução da freqüência no tempo. Aperte M para retornar ao menu principal ou F para congelar a imagem. Poderemos descongelar a imagem apertando qualquer tecla.

Se escolhermos a opção dois, o programa solicitará que iniciemos a execução da música no gravador, informando quando a gravação estiver completa e retornando então ao menu. Podemos tocar qualquer som, mas os curtos e agudos serão melhor reproduzidos. As limitações do Spectrum fazem com que o som seja reproduzido mais lentamente do que foi gravado; além disso, os sons mais graves são filtrados — a porta de entrada do gravador não é capaz de diferenciar os sons abaixo de uma certa freqüência.

É evidente que ninguém vai querer gastar quase toda a memória do seu micro simplesmente para incorporar uns poucos segundos de um som dissonante em outros programas. Contudo, se você quiser experimentar, use:

```
SAVE 'SOM' CODE 26000,39360
```

Para gravar a rotina em código que executa o som, digite:

```
SAVE 'TOCA' CODE 65440,40
```

A memória precisa ser protegida por **CLEAR 25299**. Para ouvir o som, digite **RAND USR 65440**.

Tudo isso deixará apenas 3Kbytes de memória livres para seu programa, o que não é muito, a menos que você programe em linguagem de máquina.

```
 5 CLEAR 25999: LOAD ""CODE
10 GOSUB 200: GOSUB 500
20 IF INKEY$="" THEN  GOTO 20
 21 LET C=CODE INKEY$: IF C<49
  OR C>51 THEN  GOTO 20
 22 GOSUB 30: GOSUB 200
 24 IF C=49 THEN  GOTO 100
 25 IF C=50 THEN  GOTO 600
 26 IF C=51 THEN  GOTO 700
 30 FOR N=30 TO 50 STEP 3:
 SOUND .01,N: NEXT N: RETURN
100 CLS : GOSUB 1000: PRINT AT
12,4; BRIGHT 1;"PRESSIONE QUAL
QUER TECLA"
101 IF INKEY$="" THEN  GOTO
 101
102 SOUND .1,10
104 CLS : GOSUB 150: GOSUB 800
105 FOR X=0 TO 255: PLOT X,0:
DRAW 0,USR 65368
110 IF INKEY$="m" THEN  GOSUB
30: GOTO 10
111 IF INKEY$="f" THEN  GOSUB
801: GOTO 140
130 NEXT X: CLS : GOSUB 150:
GOSUB 800: GOTO 105
140 PRINT AT 0,0;"
                    ": SOUND
.1,40: PAUSE 50: IF INKEY$=""
THEN  GOTO 140
141 SOUND .1,10: CLS : GOTO
 104
150 PRINT AT 0,2; BRIGHT 1;"
PRESSIONE (M) PARA MENU  ":
RETURN
200 BORDER 5: PAPER 5: INK 0:
CLS : RETURN
500 PRINT AT 0,2; PAPER 2; INK
7;"  MENU - ANALISADOR SONORO
 "
510 PRINT AT 5,7;"1- GRAFICO D
E BARRAS";AT 7,7;"2- GRAVAR SO
M";AT 9,7;"3- REPRODUZIR SOM"
520 PRINT AT 15,4; PAPER 4;"PR
ESSIONE (1) (2) OU (3)":
RETURN
600 PAUSE 20: GOSUB 1000:
PRINT AT 13,0;"Qualquer tecla
para GRAVAR"
605 IF INKEY$="" THEN  GOTO
 605
606 SOUND .05,20: CLS : PRINT
AT 10,8;"AGUARDE": RAND USR
65408: SOUND .1,30: CLS :
PRINT AT 10,6; BRIGHT 1;" GRAV
ACAO CONCLUIDA ": PAUSE 300:
GOTO 10
700 RAND USR 65440: GOTO 10
 800 PRINT AT 1,4; PAPER 4;"  (
 F) CONGELA A IMAGEM  ": RETURN

 801 PRINT AT 1,0; PAPER 4;" QU
 ALQUER TECLA PARA CONTINUAR ":
 RETURN
1000 PRINT AT 4,2;"Conecte o te
rminal EAR do seu";AT 6,0;"grav
ador ao terminal EAR do seu";AT
 8,0;"Spectrum. E toque alguma
musica.": RETURN
```

A rotina em código de máquina que lê a entrada do gravador encontra-se a partir da linha 1000 **DATA**. A soma dos números de cada linha está em seu trecho final para verificação, a fim de evitar que um erro de digitação acabe levando a um desastre.

Digitado o programa, grave-o em disco ou fita antes de executá-lo. Quando isto é feito, um menu aparece na tela. Se houver algum erro nas linhas **DATA**, surgirá na tela uma mensagem "VERIFIQUE A LINHA (número...)".

Aperte 1 para fazer a primeira opção. O programa solicitará que posicionemos a fita e liguemos o gravador. Ao pressionarmos a tecla **<ENTER>**, a leitura da fita começará, juntamente com o desenho do gráfico correspondente. Aperte M para retornar ao menu ou **<SHIFT><@>** para congelar a imagem.

Digite 2 para fazer a segunda opção (gravar um trecho). O programa faz as mesmas solicitações da opção 1. Enquanto a música estiver sendo executada, um gráfico colorido aparecerá na tela. Quando a gravação estiver completa, o gráfico será apagado e o programa retornará ao menu.

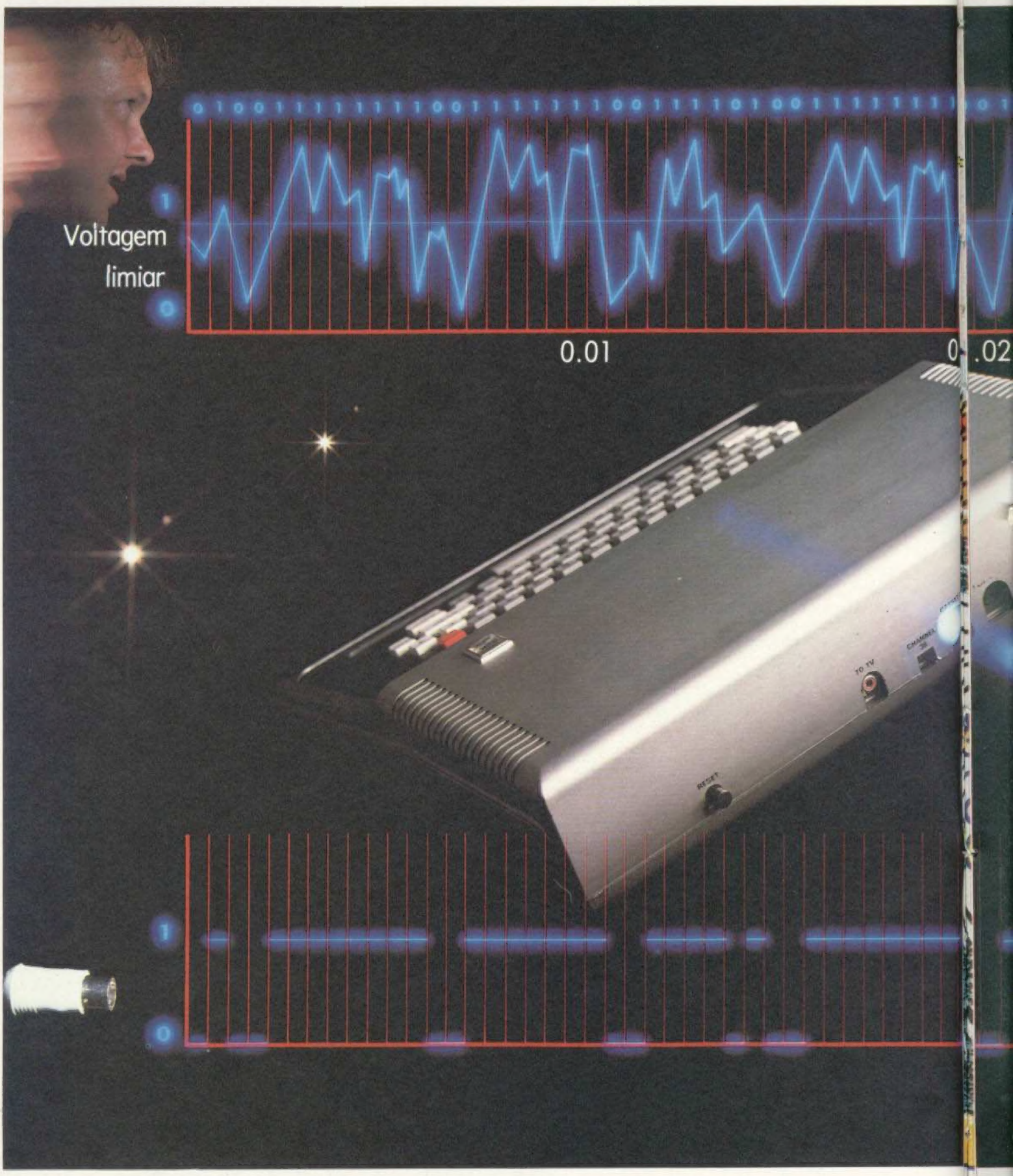
Aperte 3 para reproduzir a música

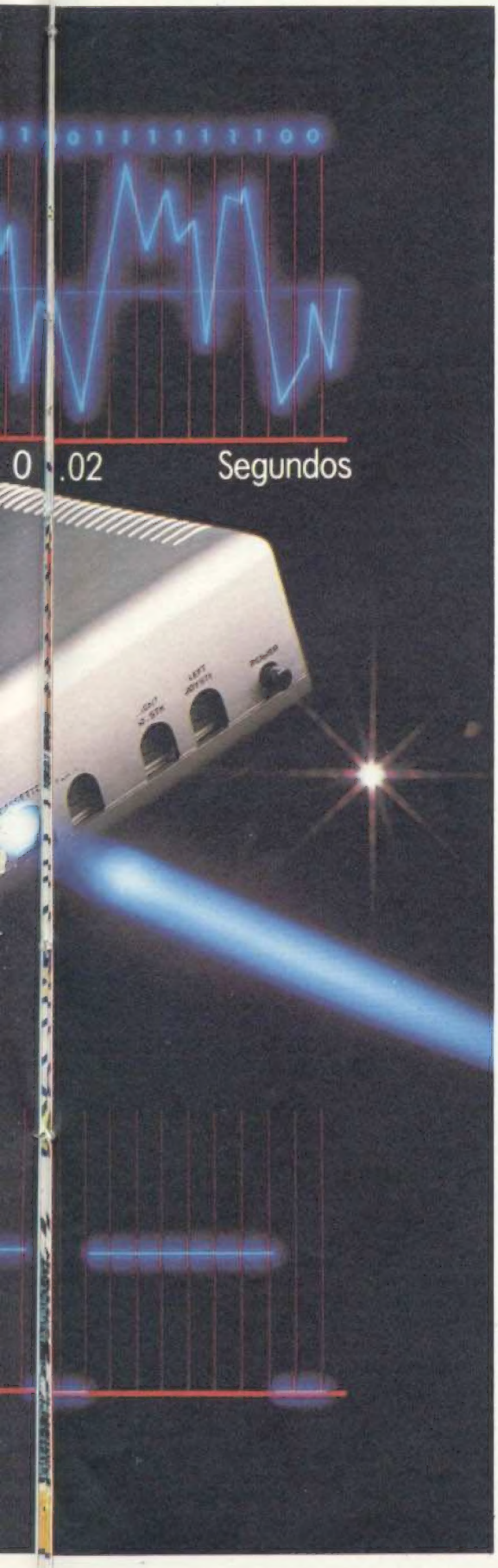
O que é análise espectral?

A análise espectral consiste na obtenção dos componentes de freqüência pura, presentes em uma determinada onda sonora. Explicando melhor: uma onda sonora complexa, como o som de uma flauta, a voz humana, o ruído de uma britadeira etc. são misturas de diversas freqüências sonoras puras. Por exemplo, em uma certa fração de tempo (medida em milissegundos), pode haver 12% de freqüência 1000 Hertz (ciclos por segundo de uma onda senoidal), 8% de 1100 Hertz etc.

O objetivo da análise espectral é quantificar cada freqüência sonora pura presente em um som. Essa quantificação, feita em termos da *potência sonora média*, geralmente é apresentada na forma de um gráfico, com a freqüência pura nas abscissas e a potência nas ordenadas.

Existem diversos algoritmos para realizar essa análise em um computador. O mais conhecido é o *Fast Fourier Transform* (FFT, Transformada Rápida de Fourier).


Voltagem
limiar
1
0
0.01
.02
TO TV
RESET



Um sinal analógico que entra pela porta do gravador cassete é lido 2000 vezes por segundo. Ao ultrapassar a voltagem limiar, o sinal é detectado e registrado como 1. Os sinais abaixo desse nível de tensão são registrados como 0. O traçado gráfico digital produzido imita a onda sonora analógica.

que acaba de gravar. Ao contrário do Spectrum, o TRS-Color executa a melodia na velocidade original. Se você quiser incorporar esse som a um outro programa, poderá gravá-lo em fita com:

```
CSAVEM'MUSICA.',1536,13823,1536
```

Se estiver usando disquetes, adicione 1536 a estes valores. Para gravar a rotina em código que executa a música, utilize:

```
CSAVEM 'TOCA',31000,31174,31091
```

É necessário proteger o topo da memória com **CLEAR 200,30099** e **PCLEAR 8.** Para executar o som, use **EXEC 31091.**

Depois de tudo isso, deve restar muito pouco espaço para seu programa; use-o economicamente.

```
10 PCLEAR 8:CLEAR 200,30999:CLS
:B=191
20 K=31000
30 READ A:IF A<0 THEN 60
40 IF A<256 THEN POKE K,A:K=K+1
:T=T+A:GOTO 30
50 IF T<>A THEN PRINT "erro  VE
RIFIQUE A LINHA";1000+10*INT((K
-31001)/20):END ELSE T=0:GOTO30
60 DEFUSR0=31000:DEFUSR1=31044:
DEFUSR2=31091
70 PRINT @14,"menu":PRINT @131,
"1- GRAFICO DE BARRAS":PRINT @1
95,"2- GRAVAR UM TRECHO":PRINT
@259,"3- REPRODUZIR MUSICA GRAV
ADA"
80 A$=INKEY$:IF A$<"1" OR A$>"3
" THEN 80
90 ON VAL(A$) GOSUB 200,400,600
100 CLS:GOTO 70
200 PMODE 4:COLOR 0,5:CLS
210 PRINT" POSICIONE O GRAVADOR
, APERTE    PLAY E TECLE <ENTER
>"
220 MOTOR ON:AUDIO ON
230 IF INKEY$<>CHR$(13) THEN 23
0
240 MOTOR OFF:PRINT @192,"PRESS
IONE 'M' PARA MENU OU       [SHI
FT] @ PARA CONGELAR IMAGEM"
250 FOR G=1TO 4000:NEXT:MOTORON
260 SCREEN 1,1
270 PCLS:FOR X=0 TO 255:A=B-4*U
SR0(0):IF A<0 THEN A=0
280 LINE (X,B)-(X,A),PSET
290 IF INKEY$="M" THEN X=255:NE
XT:MOTOR OFF:RETURN
300 NEXT:GOTO 270
```

MICRO DICAS

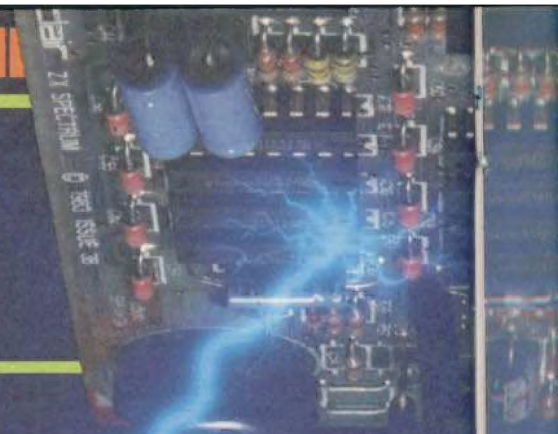
APLICAÇÃO EM JOGOS

O programa listado nesse artigo pode tornar seus jogos mais interessantes. Como ele nos permite gravar na memória a imitação de qualquer som de entrada, fica fácil usar a reprodução desse som em um jogo.

Podemos, por exemplo, acrescentar muita emoção a uma aventura introduzindo no programa ruídos de explosões, tiros e até uma mensagem curta de parabéns ao jogador. Digitalizando os sons com o auxílio de nosso programa, eles serão "tocados" pelo alto-falante do micro no momento certo. Você há de concordar que, com esse truque, a qualidade do jogo será incomparavelmente melhor.

```
400 CLS:PMODE3:MOTOR ON:AUDIO O
N:PRINT" POSICIONE O GRAVADOR,
APERTE    PLAY E TECLE <ENTER>"
410 IF INKEY$<>CHR$(13)THEN 410
420 SCREEN 1,0:N=USR1(0)
430 MOTOR OFF:RETURN
600 CLS:PRINT"  REPRODUZINDO TR
ECHO GRAVADO"
610 N=USR2(0)
620 PRINT @129,"NOVAMENTE (S/N)
 ?"
630 A$=INKEY$:IF A$<>"S" AND A$
<>"N" THEN 630
640 IF A$="S" THEN 600
650 RETURN
1000 DATA 26,80,206,255,32,142,
2,233,204,0,0,102,196,37,10,16,
163,132,48,31,1915
1010 DATA 38,245,126,180,244,19
5,0,1,32,7,102,196,36,237,16,16
3,132,48,31,38,2067
1020 DATA 245,126,180,244,26,80
,142,0,0,48,31,38,252,220,25,13
1,0,1,52,6,1847
1030 DATA 158,186,198,8,134,11,
74,38,253,118,255,32,105,132,90
,39,4,18,18,32,1903
1040 DATA 4,48,1,198,8,172,228,
38,231,53,134,26,80,182,255,1,1
32,247,183,255,2476
1050 DATA 1,182,255,3,132,247,1
83,255,3,182,255,35,138,8,183,2
55,35,158,186,220,2916
1060 DATA 25,131,0,1,52,6,134,8
,52,2,230,128,88,36,4,134,252,3
2,3,79,1397
1070 DATA 33,253,183,255,32,134
,8,74,38,253,33,251,106,228,39,
8,109,159,31,64,2291
1080 DATA 30,136,32,224,134,8,1
67,228,172,97,38,214,53,146,167
9,-1
```

INTELIGÊNCIA MILITAR



Neste último artigo da série sobre jogos de guerra, transformaremos o micro em um adversário inteligente. Veja como a heurística pode melhorar a estratégia de seu inimigo.

O jogo está completo, mas, até o momento, não oferece maiores dificuldades ao jogador. Como o computador só pode fazer movimentos aleatórios pelo tabuleiro, a superioridade do homem sobre a máquina torna-se absoluta em apenas algumas partidas. Qualquer estratégia é capaz de derrotar o computador, pois este faz seus lances independentemente dos movimentos do jogador.

Quando a vitória fácil começa a nos entediar, a saída é fortalecer nosso oponente. Isso significa incluir mais rotinas. Como a principal dificuldade na programação de jogos do tipo *Capa e Espada* é exatamente a limitação da memória — sobretudo nos micros das linhas Apple e TK-2000 —, tudo o que se adicionar ao programa deve reunir simplicidade e eficácia.

UM INIMIGO MAIS FORTE

Há duas maneiras de tornar o computador um adversário mais forte. Uma delas consiste em reconhecer sua limitação intelectual dando-lhe unidades superiores, em força, às do jogador. A maioria dos jogos comerciais adota essa solução, que é, sem dúvida, a de programação mais fácil.

Para observar o efeito de tal mudança, basta que se adicione uma linha ao programa. Você logo perceberá que esta não é realmente a saída ideal. Contudo, não custa experimentá-la: seu único trabalho será introduzir — e depois apagar — a linha que dá a "força extra" ao computador.

O elemento da matriz da tropa que contém a força inicial da unidade é o 6. Para aumentá-lo, digite:

```
665 IF J=0 THEN  LET T(J+1,6)=
T(J+1,6)+FN R(100): LET T(J+1,
7)=T(J+1,6)
```

```
665 IF J=8 THEN T(J+I,6)=T(J+I,
6)+RND(100):T(J+I,7)=T(J+I,6)
```

```
665  IF J = 8 THEN T(J + 1,6) =
 T(J + 1,6) +  FN R(100):T(J +
I,7) = T(J + I,6)
```

```
665 IF J=8 THEN T(J+I,6)=T(J+I,
6)+RND(100):T(J+I,7)=T(J+I,6)
```

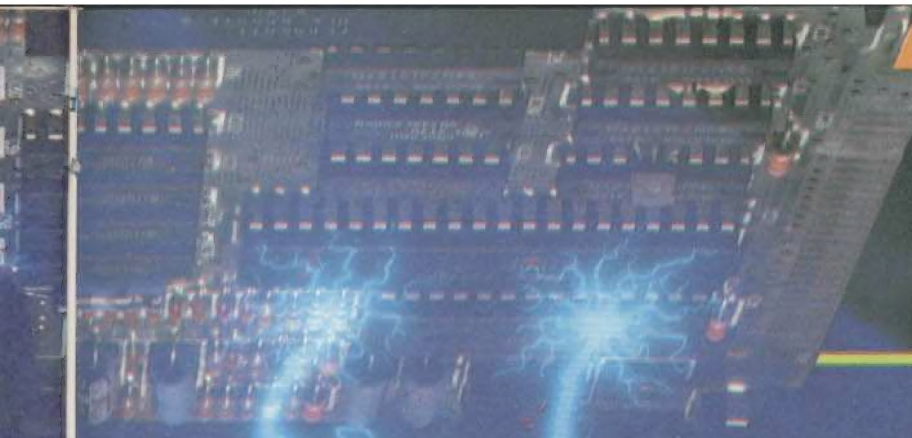
O programa somará um número randômico ao poder inicial da unidade, colocando o resultado no seu poder atual.

INTELIGÊNCIA MILITAR

Um oponente com forças iguais e, ainda, inteligente será bem mais interessante do que um inimigo forte demais e intelectualmente incapaz. Contudo, aumentar a inteligência é bem mais difícil que aumentar a força.

Os conceitos utilizados na programação da inteligência em *Capa e Espada* são bem diferentes daqueles que vimos em *Otelo* ou em *A Raposa e os Gansos*.

Nesses jogos de tabuleiro, os movimentos são muito bem definidos. Em ambos é possível prever movimentos futuros bem como critérios exatos de sucesso, utilizando a pesquisa em árvore e outros processos mais simples. Além disso, os algoritmos utilizados nos programas não envolvem elementos de acaso. No caso de *A Raposa e os Gansos*,



- UM INIMIGO HABILIDOSO
- INTELIGÊNCIA X FORÇA BRUTA
- USE A HEURÍSTICA
- ESTRATÉGIAS SUTIS
- UMA NOVA ROTINA

o algoritmo é muito eficiente nos níveis de dificuldade mais altos, tornando muito custosa a vitória do jogador. Em *Otelo*, o algoritmo é mais simples, oferecendo menos dificuldade.

Nos jogos de guerra é quase impossível definir um algoritmo. Não há movimentos determinados para nenhum dos lados — e, nesse aspecto, a diferença entre jogos de guerra e xadrez vem à tona. No xadrez não existe o acaso e todos os elementos envolvidos estão relacionados aos movimentos e posições no tabuleiro. Os jogos de guerra, ao contrário, incluem vários elementos aleatórios e exigem a consideração de numerosas variáveis — armadura, moral, capacidade de movimento, poder etc. Algumas dessas variáveis são inter-

dependentes, outras não. E, acima de tudo, não há receita para a vitória: nenhum procedimento leva inevitavelmente ao triunfo.

Poderíamos examinar a possibilidade de definir um algoritmo eficaz para o nosso jogo. Mas este seria, de qualquer maneira, extremamente complexo, e tornaria o programa grande e lento demais. Na prática, a saída para programas grandes, complicados e sem algoritmos definidos é dada por um conjunto de *heurísticas*.

Uma heurística é apenas uma regra prática que parece funcionar na maioria dos casos. Não há garantias de que ela funcione nem de que não leve a erros brutais em determinadas situações. Geralmente, no entanto, vale a pena tentar; afinal, este é o procedimento da maior parte das pessoas diante de muitos problemas complicados.

O programa conterá, então, uma lista de regras práticas, a saber:

• Se você é uma unidade de arqueiros e existe uma unidade inimiga ao seu alcance, atire.

• Se você se envolver em uma batalha e estiver perdendo, tente ir em direção contrária.

• Se você está perto de uma unidade inimiga mais forte, afaste-se.

• Se você está próximo a uma unidade inimiga mais fraca, aproxime-se.

No último caso, o computador leva vantagem, já que conhece o poder das unidades do jogador. Este, ao contrário, não tem acesso à mesma informação sobre as tropas inimigas.

CONDUTA PLANEJADA

Além dessas regras, o computador precisa de um plano geral de conduta. O desenrolar de uma guerra nunca é deixado ao acaso. O plano dependerá das condições de vitória — se esta depende, por exemplo, da morte do comandante, poderá ser conveniente usar todas as forças contra o quartel-general. Como em *Capa e Espada* a vitória depende de causar mais baixas ao inimigo do que sofrê-las, o plano do computador terá esse objetivo.

Um modo simples de eliminar as tropas inimigas consistiria em atacar unidades fracas com unidades fortes. Mas esta seria uma estratégia difícil de programar. Ela poderia ser simplificada para "concentre todas as suas forças numa posição". Fazendo isso, as forças do computador superariam as do jogador — desde que este não tivesse planejado a mesma coisa. Essa estratégia, porém, tem uma conseqüência: as forças do jogador superariam as do computador em outra parte do tabuleiro — e aqui a heurística falharia.

Para que o jogador não possa prever a ação do inimigo, o computador deve ter algumas opções abertas. Novamente preservando a simplicidade, podemos fazer a máquina escolher sempre, para concentrar suas forças, uma das posições ocupadas por unidades do jogador. Porém, uma rotina verificaria essas posições, transferindo a concentração ao acaso, a cada volta do laço principal. Isso manteria o jogador na incerteza, ao mesmo tempo que o plano do computador seguiria seu curso.

Um segundo aspecto do jogo a ser inteligentemente controlado é a ação individual das unidades. Cada uma delas deve ser capaz de responder às condições locais do campo de batalha, independentemente do plano geral. Uma unidade não deve, por exemplo, dirigir-se ao ponto de concentração se há unidades inimigas mais fortes no caminho.

Quando o computador tiver que dar ordens, ele fará uma série de testes, numa seqüência cuidadosamente escolhida. Essa escolha obedece a certas regras, elaboradas conforme dois critérios. Primeiro: como os testes precisam ser executados com a maior rapidez possível, os que não se aplicam à situação devem ser descartados. Segundo: os testes mais importantes serão realizados no início, de maneira a impedir que uma decisão seja tomada com base em um fator menos relevante.

As regras usadas no programa são as seguintes, por ordem de importância:

• Se a unidade está em combate e venceu na última vez, as ordens são mantidas. Ignoram-se as outras condições.

• Se a unidade está em combate e perdeu na última vez, deve se afastar do inimigo. Isso nem sempre é possível, pois ela pode estar na borda do mapa ou ter o caminho de fuga obstruído.

• Se uma unidade de arqueiros tem um alvo ao alcance, ela deve atirar. A posição dessa regra assegura que os arqueiros prefiram atirar a se envolver em combate corpo a corpo.

• Se existe uma unidade inimiga mais fraca a uma distância inferior ao deslocamento máximo, MARCHE em direção a ela. Entretanto, se a unidade for mais forte, afaste-se. Este é um teste demorado, uma vez que leva em conta as unidades inimigas.

• Dirija-se ao ponto de concentração.

Mesmo essas poucas regras levam um bom tempo para serem consideradas, diminuindo bastante a velocidade do jogo. Este é o preço a ser pago por um jogo inteligente. Não haverá, porém, esperas tão longas e exasperantes como em *A Raposa e os Gansos*.

ROTINAS ADICIONAIS

Apague as linhas 1770 a 1790 antes de adicionar estas rotinas.

```
 360 DIM t$(8,12): DIM o$(5,12)
 : DIM w$(5,9): DIM m$(5,12):
 DIM a$(4,12): DIM r$(4,12):
 DIM c(8)
 416 LET sp=1
1665 IF wn>8 THEN  LET c(wn-8)=
8
1666 IF lo>8 THEN  LET c(lo-8)=
T(wn,2)
1760 LET ra=st: LET rb=sh: LET
rc=fx: LET rd=fy: GOSUB 3000
2140 REM Inimigo
2142 REM Loop
2143 LET r=FN r(10)
2144 IF r=1 OR T(sp,1)>3 THEN
LET sp=FN r(8)
2145 IF r=1 AND T(sp,1)>3 THEN
 GOTO 2142
2150 IF c(e-8)=8 THEN  RETURN
2155 IF c(e-8)<>0 THEN  LET T(e
,1)=3: LET T(e,2)=c(e-8): LET c
(e-8)=0: RETURN
2170 IF T(e,3)=2 THEN  LET ra=1
: LET rb=e: LET rc=5: LET rd=5:
 GOSUB 3000: IF gp<>-1 THEN  LE
T T(e,1)=1: RETURN
2180 LET T(e,1)=3
2181 LET hp=5: LET vp=5: LET mv
=0
2182 FOR v=1 TO 8
2183 LET zp=0: GOSUB 3100
2184 IF zp<>0 THEN  GOSUB 3200
2185 NEXT v
2187 IF hp<>5 AND vp<>5 THEN  R
ETURN
2188 IF mv<>0 THEN  LET T(e,2)=
mv: RETURN
2189 LET hp=T(e,8)-T(sp,8): LET
 vp=T(e,9)-T(sp,9): GOSUB 3200
2190 RETURN
3000 REM
3010 LET gp=-1
3020 FOR m=ra TO (ra+7)
3030 LET xx=ABS (T(m,8)-T(rb,8)
): LET yy=ABS (T(m,9)-T(rb,9))
3040 IF xx<rc AND yy<rd AND T(m
,1)<4 THEN  LET rc=xx: LET rd=y
y: LET gp=m
```

```
3050 NEXT m
3060 RETURN
3100 REM
3110 IF T(v,1)>3 THEN  RETURN
3120 LET xx=ABS (T(v,8)-T(e,8))
: LET yy=ABS (T(v,9)-T(e,9))
3130 IF T(v,7)>=T(e,7) AND xx<5
 AND yy<5 THEN  LET mv=T(v,2)
3140 IF xx<hp AND yy<vp THEN  L
ET hp=xx: LET vp=yy: LET zp=1:
RETURN
3150 RETURN
3200 REM
3210 IF hp>=0 THEN  LET lp=1
3220 IF hp<0 THEN  LET lp=3: LE
T hp=ABS (hp)
3230 IF vp>=0 AND ABS (vp)>hp T
HEN  LET lp=2
3240 IF vp<0 AND ABS (vp)>hp TH
EN  LET lp=4: LET vp=ABS (vp)
3250 LET T(e,2)=lp
3260 RETURN
```

```
'360 DIM T$(8),O$(5),W$(5),M$(5)
,A$(4),R$(4),C(8)
416 SP=1
1665 IF WN>8 THEN C(WN-8)=8
1666 IF LO>8 THEN C(LO-8)=T(WN,
2)
1760 RA=ST:RB=SH:RC=FX:RD=FY:GO
SUB 3000
2140 REM  INIMIGO
2142 REM  LOOP
2143 R=RND(10)
2144 IF R=1 OR T(SP,1)>3 THEN S
P=RND(8)
2145 IF R=1 AND T(SP,1)>3 THEN
2142
2150 IF C(E-8)=8 THEN RETURN
2155 IF C(E-8)<>0 THEN T(E,1)=3
:T(E,2)=C(E-8):C(E-8)=0:RETURN
2170 IF T(E,3)=2 THEN RA=1:RB=E
:RC=5:RD=5:GOSUB 3000:IF GP<>-1
 THEN T(E,1)=1:RETURN
2180 T(E,1)=3
2181 HP=5:VP=5:MV=0
2182 FOR V=1 TO 8
2183 ZP=0:GOSUB 3100
2184 IF ZP<>0 THEN GOSUB 3200
2185 NEXT V
2187 IF HP<>5 AND VP<>5 THEN RE
TURN
2188 IF MV<>0 THEN T(E,2)=MV:RE
TURN
2189 HP=T(E,8)-T(SP,8):VP=T(E,9
)-T(SP,9):GOSUB 3200
2190 RETURN
3000 REM
3010 GP=-1
3020 FOR M=RA TO (RA+7)
3030 XX=ABS(T(M,8)-T(RB,8)):YY=
ABS(T(M,9)-T(RB,9))
3040 IF XX<RC AND YY<RD AND T(M
,1)<4 THEN RC=XX:RD=YY:GP=M
3050 NEXT M
3060 RETURN
3100 REM
3110 IF T(V,1)>3 THEN RETURN
3120 XX=ABS(T(V,8)-T(E,8)):YY=A.
```

```
BS(T(V,9)-T(E,9))
3130 IF T(V,7)>=T(E,7) AND XX<5
 AND YY<5 THEN MV=T(V,2)
3140 IF XX<HP AND YY<VP THEN HP
=XX:VP=YY:ZP=1:RETURN
3150 RETURN
3200 REM
3210 IF HP>=0 THEN LP=1
3220 IF HP<0 THEN LP=3:HP=ABS(H
P)
3230 IF VP>=0 AND ABS(VP)>HP TH
EN LP=2
3240 IF VP<0 AND ABS(VP)>HP THE
N LP=4:VP=ABS(VP)
3250 T(E,2)=LP
3260 RETURN
```

```
360  DIM T$(8),O$(5),W$(5),M$(5
),A$(4),R$(4),C(8)
416 SP = 0
1665  IF WN > 8 THEN C(WN - 8)
= 8
1666  IF LO > 8 THEN C(LO - 8)
= T(WN,2)
1760 RA = ST:RB = SH:RC = FX:RD
 = FY: GOSUB 3000
2140  REM  SELECAO
2143 R =  FN R(10)
2144  IF R = 1 OR T(SP,1) > 3 T
HEN SP =  FN R(8)
2145  IF R = 1 AND T(SP,1) > 3
THEN 2143
2150  IF C(E - 8) = 8 THEN  RET
URN
2155  IF C(E - 8) <  > 0 THEN T
(E,1) = 3:T(E,2) = C(E - 8):C(E
 - 8) = 0: RETURN
2170  IF T(E,3) = 2 THEN RA = 1
:RB = E:RC = 5:RD = 5: GOSUB 30
00: IF GP <  >  - 1 THEN T(E,1)
 = 1: RETURN
2180 T(E,1) = 3
2181 HP = 5:VP = 5:MV = 0
2182  FOR V = 1 TO 8
2183 ZP = 0: GOSUB 3100
2184  IF ZP <  > 0 THEN  THEN
GOSUB 3200
2185  NEXT V
2187  IF HP <  > 5 AND VP <  >
5 THEN  RETURN
2188  IF MV <  > 0 THEN T(E,2)
= MV: RETURN
2189 HP = T(E,8) - T(SP,8):VP =
 T(E,9) - T(SP,9): GOSUB 3200
2190  RETURN
3000  REM  ALCANCE
3010 GP =  - 1
3020  FOR M = RA TO (RA + 7)
3030 XX =  ABS (T(M,8) - T(RB,8
)):YY =  ABS (T(M,9) - T(RB,9))
3040  IF XX < RC AND YY < RD AN
D T(M,1) < 4 THEN RC = XX:RD =
YY:GP = M
3050  NEXT M
3060  RETURN
3100  REM  PODER
3110  IF T(V,1) > 3 THEN  RETUR
N
3120 XX =  ABS (T(V,8) - T(E,8)
):YY = T(V,9) - T(E,9))
3130  IF T(V,7) >  = T(E,7) AND
 XX < 5 AND YY < 5 THEN MV = T(
V,2)
3140  IF XX < HP AND YY < VP TH
EN HP = XX:VP = YY:ZP = 1: RETU
RN
3150  RETURN
3200  REM  CONCENTRACAO
3210  IF HP >  = 0 THEN LP = 1
3220  IF HP < 0 THEN LP = 3:HP
=  ABS (HP)
3230  IF VP >  = 0 AND  ABS (VP
) > HP THEN LP = 2
3240  IF VP < 0 AND  ABS (VP) >
 HP THEN LP = 4:VP =  ABS (VP)
3250 T(E,2) = LP
3260  RETURN
```

As rotinas que começam em 3000 devem ser renumeradas a fim de liberar espaço para essas linhas. Digite:

```
RENUM 4000,3000
```

Em seguida, acrescente estas linhas:

```
360 DIM T$(8),O$(5),W$(5),M$(5)
,A$(4),R$(4),C(8)
416 SP=1
1665 IF WN>8 THEN C(WN-8)=8
1666 IF LO>8 THEN C(LO-8)=T(WN,
2)
1760 RA=ST:RB=SH:RC=FX:RD=FY:GO
SUB 3000
2140 REM  INIMIGO
2142 REM  LOOP
2143 R=RND(10)
2144 IF R=1 OR T(SP,1)>3 THEN S
P=RND(8)
2145 IF R=1 AND T(SP,1)>3 THEN
2142
2150 IF C(E-8)=8 THEN RETURN
2155 IF C(E-8)<>0 THEN T(E,1)=3
:T(E,2)=C(E-8):C(E-8)=0:RETURN
2170 IF T(E,3)=2 THEN RA=1:RB=E
:RC=5:RD=5:GOSUB 3000:IF GP<>-1
 THEN T(E,1)=1:RETURN
2180 T(E,1)=3
2181 HP=5:VP=5:MV=0
2182 FOR V=1 TO 8
2183 ZP=0:GOSUB 3100
2184 IF ZP<>0 THEN GOSUB 3200
2185 NEXT V
2187 IF HP<>5 AND VP<>5 THEN RE
TURN
2188 IF MV<>0 THEN T(E,2)=MV:RE
TURN
2189 HP=T(E,8)-T(SP,8):VP=T(E,9
)-T(SP,9):GOSUB 3200
2190 RETURN
3000 REM
3010 GP=-1
3020 FOR M=RA TO (RA+7)
3030 XX=ABS(T(M,8)-T(RB,8)):YY=
ABS(T(M,9)-T(RB,9))
3040 IF XX<RC AND YY<RD AND T(M
,1)<4 THEN RC=XX:RD=YY:GP=M
3050 NEXT M
3060 RETURN
3100 REM
3110 IF T(V,1)>3 THEN RETURN
3120 XX=ABS(T(V,8)-T(E,8)):YY=A
BS(T(V,9)-T(E,9))
3130 IF T(V,7)>=T(E,7) AND XX<5
 AND YY<5 THEN MV=T(V,2)
3140 IF XX<HP AND YY<VP THEN HP
=XX:VP=YY:ZP=1:RETURN
3150 RETURN
3200 REM
3210 IF HP>=0 THEN LP=1
3220 IF HP<0 THEN LP=3:HP=ABS(H
P)
3230 IF VP>=0 AND ABS(VP)>HP TH
EN LP=2
3240 IF VP<0 AND ABS(VP)>HP THE
N LP=4:VP=ABS(VP)
3250 T(E,2)=LP
3260 RETURN
```

COMO FUNCIONA

As adições anteriores à linha 2140 cuidam de algumas variáveis extras que serão utilizadas.

No início da rotina de escolha, há uma chance em dez de que o computador mude o ponto de concentração. Este é verificado e modificado de acordo, mas não se realiza nenhuma ação neste estágio — outros testes precisam ainda ser feitos.

Se a unidade for constituída de arqueiros, a linha 2170 usa a rotina de alcance para decidir se atira. Caso os arqueiros não disparem, ou se a unidade for de outro tipo, a linha 2180 muda a ordem para MARCHE. A seção seguinte do programa verifica cada uma das unidades inimigas. A linha 2183 utiliza a rotina de poder para determinar a direção do movimento.

Se a unidade em questão não estiver envolvida em combate nem prestes a se envolver, a rotina de concentração (linha 3500) faz com que ela se mova em direção ao ponto de concentração.

A ESCOLHA É SUA

Você pode adicionar qualquer das versões aqui apresentadas, ou mesmo as duas, para melhorar o desempenho do micro, enquanto jogador. Tudo dependerá do tipo de oponente que desejar.

A própria natureza das heurísticas incorporadas ao programa faz com que elas falhem em certas circunstâncias. Alguns jogadores acharão essas régras menos adequadas do que outras eventualmente deduzidas pela prática. Só um processo de tentativa e erro determinará as melhores estratégias e muitas outras rotinas poderão ser tentadas.

NOVAS MENSAGENS SECRETAS

- COMO DECIFRAR CÓDIGOS
- PROGRAMA DE DISTRIBUIÇÃO DE LETRAS
- CÓDIGOS MULTIPLICATIVOS
- LIVRO-CÓDIGO

Neste artigo você ficará conhecendo novas maneiras de transmitir mensagens confidenciais, sem correr o risco de que elas sejam decifradas por pessoas indesejadas.

No artigo *Mensagens Secretas* (página 888), sugerimos vários métodos para a transmissão de informações de caráter confidencial. Alguns deles são relativamente fáceis de serem decifrados; entretanto, se recorrermos ao auxílio do computador, poderemos chegar a métodos mais complexos e com diferentes níveis de sofisticação.

COMO DECIFRAR CÓDIGOS

Da mesma maneira que os criptógrafos tentam desenvolver códigos que ofereçam mais segurança, muitos especialistas estudam uma forma de frustrar essas tentativas. Uma arma muito poderosa para se decifrar simples códigos de substituição ou de transposição consiste na contagem de freqüência das letras. Em português, as letras que ocorrem com mais freqüência são — nesta ordem — A, E, O, S, R, I, C. Desse modo, se em um texto codificado aparecerem muitas vezes as letras AEOSRIC, provavelmente você estará manipulando um código de transposição. Se outras letras se repetirem mais vezes, a codificação pode ter sido feita com um método de substituição. É importante levar em conta que essa distribuição é exclusiva de cada idioma. Portanto, torna-se fundamental saber em que língua foi escrita a mensagem.

Seja como for, será necessário fazer a contagem da freqüência das letras na mensagem. Isso pode consumir muito tempo e é um processo bastante susceptível de erros. O programa aqui apresentado lhe será útil nesses casos. Basta que você digite o texto e o computador imediatamente o informará do número de vezes que cada letra apareceu.

A figura da página 1091 mostra a freqüência das letras em um texto consti-

tuído de cem palavras. Observe que os números seguem, razoavelmente, o princípio que acabamos de comentar.

Agora, digite o programa e veja como isso funciona na prática.

```
15 POKE 23658,8
20 BORDER 0: PAPER 0: INK 7:
CLS
30 PRINT TAB 5;"CONTAGEM DA F
REQUENCIA"''
40 PRINT TAB 12;"CUIDADO"''
50 PRINT  FLASH 1;AT 4,7;"NAO
  DEIXE ESPACOS";AT 5,9;"ENTR
E PALAVRAS"''
60 DIM n(28)
70 PRINT " Para finalizar ent
rada do texto digite '*'"
80 FOR t=1 TO 28: LET n(t)=0:
 NEXT t
90 INPUT "Introduza o texto";
a$
100 IF a$="*" THEN  GOTO 180
110 CLS
120 FOR i=1 TO LEN a$
130 FOR j=1 TO 26
140 IF j=CODE (a$(i TO i))-64
THEN  LET n(j)=n(j)+1
150 NEXT j
160 NEXT i
170 GOTO 90
180 CLS
190 PRINT "Letra    Freq'   Le
tra   Freq'"
200 FOR i=1 TO 13
210 PRINT TAB 2;CHR$ (64+i);
TAB 10;n(i);TAB 19;CHR$ (77+i)
;TAB 27;n(13+i)
220 NEXT i
230 STOP
```

```
20 CLS
30 PRINT @5,"CONTAGEM DA FREQUE
NCIA"
40 PRINT @76,"CUIDADO"
50 PRINT @134,"NAO DEIXE ESPACO
S":PRINT @166,"ENTRE AS PALAVRA
S"
60 DIM N(28)
70 PRINT @225,"PARA FINALIZAR E
NTRADA DO TEXTO E EMITIR RESULT
ADOS DIGITE *"
80 FOR T=1 TO 28:N(T)=0:NEXT
90 INPUT"INTRODUZA O TEXTO ";A$
100 IF A$="*" THEN 180
110 CLS
120 FOR I=1 TO LEN(A$)
130 FOR J=1 TO 26
140 IF J=ASC(MID$(A$,I,1))-64 T
HEN N(J)=N(J)+1
150 NEXT J
160 NEXT I
170 GOTO 90
180 CLS
190 PRINT "LETRA    FREQ'   LET
RA    FREQ'"
200 FOR I=1 TO 13
210 PRINT TAB(2);CHR$(64+I);TAB
(10);N(I);TAB(19);CHR$(77+I);TA
B(27);N(13+I)
220 NEXT
230 END
```

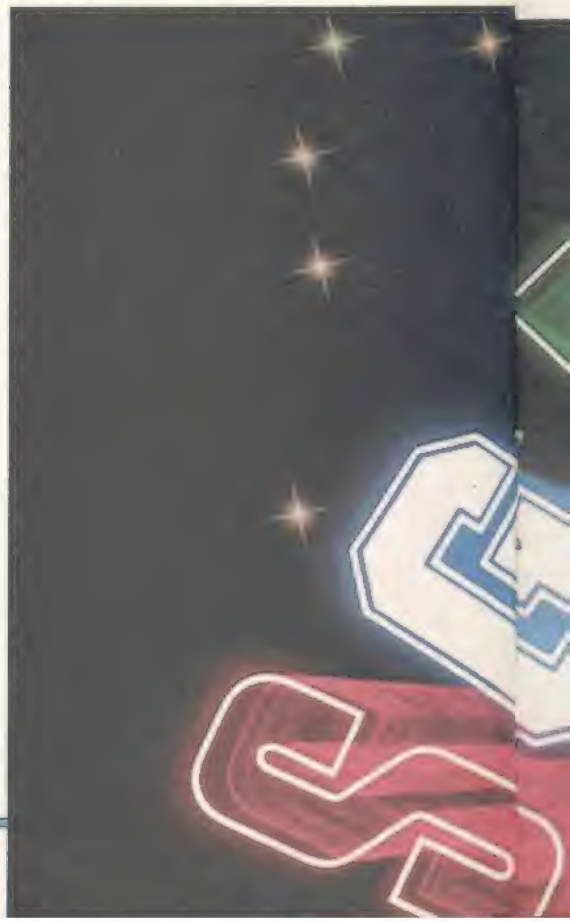
A tabela de freqüência das letras tem grande utilidade na decifração de códigos.

```
20  HOME
30  PRINT  TAB( 9)"CONTAGEM DE
FREQUENCIA": PRINT
40  PRINT  TAB( 17)"CUIDADO": P
RINT
50  PRINT  TAB( 12)"NAO DEIXE E
SPACOS": PRINT  TAB( 12)"ENTRE
AS PALAVRAS"
60  DIM N(28)
70  PRINT : PRINT : PRINT  TAB(
 8)"PARA FINALIZAR A ENTRADA":
PRINT  TAB( 11)" DO TEXTO DIGIT
E *"
80  FOR I = 1 TO 28:N(T) = 0: N
EXT
90  PRINT : INPUT "DIGITE O TEX
TO ";A$
100  IF A$ = "*" THEN 180
110  HOME
120  FOR I = 1 TO  LEN (A$)
130  FOR J = 1 TO 26
140  IF J =  ASC ( MID$ (A$,I,1
)) - 64 THEN N(J) = N(J) + 1
150  NEXT J
160  NEXT I
170  GOTO 90
180  HOME
190  PRINT "LETRA   FREQ   LETR
A   FREQ"
200  FOR I = 1 TO 13
210  PRINT  TAB( 2); CHR$ (64 +
 I); TAB( 10);N(I); TAB( 17); C
HR$ (77 + I); TAB( 25);N(13 + I
)
220  NEXT I
230  END
```

```
20 CLS
30 PRINT TAB(9)"CONTAGEM DE FRE
QUENCIA":PRINT
40 PRINT TAB(17)"CUIDADO":PRINT
50 PRINT TAB(12)"NAO DEIXE ESPA
ÇOS":PRINT TAB(12)"ENTRE AS PAL
AVRAS"
60 DIM N(28)
70 PRINT:PRINT:PRINT TAB(8)"PAR
A FINALIZAR A ENTRADA":PRINT TA
B(12)"DO TEXTO DIGITE *"
80 FOR I=1 TO 28:N(T)=0:NEXT
90 PRINT:INPUT "DIGITE O TEXTO"
;A$
100 IF A$="*" THEN 180
110 CLS
120 FOR I=1 TO LEN(A$)
130 FOR J=1 TO 26
140 IF J=ASC(MID$(A$,I,1))-64 T
HEN N(J)=N(J)+1
150 NEXT J
160 NEXT I
170 GOTO 90
180 CLS
190 PRINT "LETRA   FREQ   LETRA
   FREQ"
200 FOR I=1 TO 13
210 PRINT TAB(2);CHR$(64+I);TAB
(10);N(I);TAB(17);CHR$(77+I);TA
B(25);N(13+I)
220 NEXT I
230 END
```

A estrutura operacional deste programa está fundamentada em um simples mecanismo de contagem.

Em sua primeira parte é atribuído o valor zero às 28 variáveis indexadas, sendo que 26 dessas variáveis serão usadas para a contagem da freqüência com que aparecem as 26 letras. As duas variáveis restantes foram incluídas tendo em vista a possibilidade de uma futura ampliação do programa; nesse caso, podem ser incorporados algarismos ou quaisquer outros sinais gráficos.



CÓDIGOS MULTIPLICATIVOS

Durante a Guerra Civil Norte-Americana (1861-1865) criou-se um tipo especial de código para a comunicação entre as tropas, que funciona da seguinte forma: suponhamos que se queira passar para um oficial preso a mensagem ESCAPAR PARA LONDRES. Nesse caso, a primeira letra da frase seria colocada na primeira linha, a segunda letra na segunda linha, a terceira letra de volta na primeira linha e assim por diante, de tal maneira que a frase acabaria ficando assim:

E C P R A A O D E
S A A P R L N R S

Escrita por extenso, a mensagem seria a seguinte: ECPRAAODESAAPRLNRS; dividida a fim de confundir o inimigo: ECPRAA OD ESAAP RLNRS.

Na verdade, esse código é um caso especial do que se chama atualmente código multiplicativo.

Em nossa mensagem, o texto contém dezoito caracteres. Estes podem ser arranjados em matrizes do tipo: 2 x 9, 9 x 2, 3 x 6, ou 6 x 3, como mostram as figuras da página 1095.

Qualquer pessoa que queira decifrar a mensagem, sem saber que se trata de um código multiplicativo, terá muito trabalho. O processo de codificação, no entanto, é muito simples: tudo o que se tem a fazer é escrever a mensagem verticalmente, letra por letra, preenchendo totalmente a primeira coluna da matriz. Em seguida, partimos para a próxima coluna e repetimos o processo, até que a mensagem termine e a matriz esteja completa. Finalmente, copiamos as linhas lado a lado, começando sempre pela primeira.

É importante que a mensagem ocupe totalmente a matriz escolhida. Para tanto, podemos introduzir no texto alguns caracteres ou palavras sem nexo, que terão a função extra de confundir quem tentar decifrá-lo.

O laço que se encontra entre as linhas 140 e 210 é a parte principal do programa, responsável tanto pela codificação como pela decodificação.

```
 20 BORDER 0: PAPER 0: INK 7:
 CLS
 30 PRINT TAB (6);"CODIGO MULT
 IPLICATIVO"
 40 PRINT : PRINT : PRINT
 50 PRINT  FLASH 1; PAPER 2;"N
 AO DEIXE ESPACOS ENTRE PALAVR
 AS"
 60 INPUT "INTRODUZA O TEXTO "
 'm$
 70 INPUT "LINHAS ?  ";m
 80 INPUT "COLUNAS ? ";n
 90 INPUT "(c)ODIFICAR OU (d)E
 CODIFICAR ? ";e$
100 PAUSE 50: CLS
110 IF e$="c" THEN  LET x=m
120 IF e$="d" THEN  LET x=n
130 DIM d$(x,LEN m$/x)
140 FOR i=1 TO x
155 LET a$="": LET m$=m$+" "
160 FOR j=1 TO LEN m$-1 STEP x
180 LET a$=a$+m$(i+j-1 TO i+j-
1)
190 NEXT j
195 LET d$(i)=a$
197 LET m$=m$( TO LEN m$-1)
200 PRINT d$(i);: IF e$="c"
THEN  PRINT " ";
210 NEXT i
220 STOP
```

```
20 CLS
30 PRINT @6,"CODIGO MULTIPLICAT
IVO"
40 PRINT:PRINT:PRINT
50 PRINT"NAO DEIXE ESPACOS ENTR
E PALAVRAS"
60 PRINT:INPUT"TEXTO  ";M$
70 INPUT"LINHAS ";M
80 INPUT"COLUNAS ";N
90 INPUT"(C)ODIFICAR OU (D)ECOD
IFICAR ";E$
100 FOR L=1 TO 1000:NEXT
110 IF E$="C" THEN X=M
120 IF E$="D" THEN X=N
130 DIM D$(X)
140 FOR I=1 TO X
150 D$(I)=" "
160 FOR J=1 TO LEN(M$) STEP X
170 B$=MID$(M$,I+J-1,1)
180 D$(I)=D$(I)+B$
190 NEXT J
200 PRINT D$(I)
210 NEXT I
220 END
```

Há vantagens em se criptografar um programa de computador?

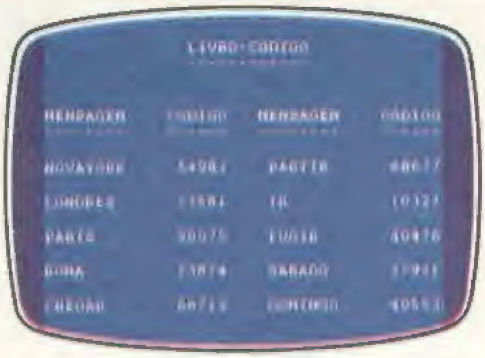
A resposta depende do tipo de programa. Não há nenhuma vantagem em criptografar um programa em linguagem de máquina (código binário) — tarefa, aliás, impossível para muitos micros, que não poderão executá-lo depois. Mas, se o programa estiver em código-fonte (linguagem de alto nível, como BASIC), às vezes pode ser vantajoso criptografá-lo.

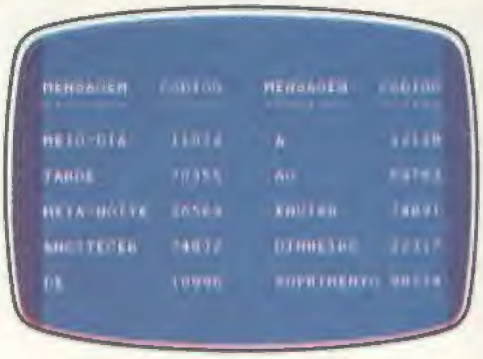
O objetivo da criptografia de programas é protegê-los contra a cópia e a imitação ilegais. Uma aplicação muito freqüente ocorre na transmissão de programas de computador em sistemas telemáticos. Aqui, o objetivo é torná-los imunes à cópia ilegal no momento da transmissão. Nesse caso, o interessado pode criptografar o programa como um texto qualquer, usando um dos métodos explicados neste artigo (o multiplicativo costuma ser o mais empregado).

```
20  HOME
30  PRINT  TAB( 10)"CODIGO MULT
IPLICATIVO"
40  PRINT : PRINT : PRINT
50  PRINT  TAB( 12)"NAO DEIXE E
SPACOS": PRINT  TAB( 12)"ENTRE
AS PALAVRAS"
60  PRINT : INPUT "TEXTO:";M$
70  INPUT "LINHAS:";M
80  INPUT "COLUNAS:";N
90  INPUT "CODIFICAR(C) OU DECO
DIFICAR(D):";E$
100  FOR L = 1 TO 1000: NEXT
110  IF E$ = "C" THEN X = M
120  IF E$ = "D" THEN X = N
130  DIM D$(X)
140  FOR I = 1 TO X
150  IF E$ = "C" THEN D$(I) = "
 "
160  FOR J = 1 TO  LEN (M$) STE
P X
170 B$ =  MID$ (M$,I + J - 1,1)

180 D$(I) = D$(I) + B$
190  NEXT J
200  PRINT D$(I);: IF E$ = "C"
THEN  PRINT
210  NEXT I
220  END
```

```
20 CLS
30 PRINT TAB(10)"CODIGO MULTIPL
ICATIVO"
40 PRINT:PRINT:PRINT
50 PRINT TAB(12)"NAO DEIXE ESPA
ÇOS":PRINT TAB(12)"ENTRE AS PAL
AVRAS"
60 PRINT:INPUT"TEXTO:";M$
70 INPUT"LINHAS:";M
80 INPUT"COLUNAS:";N
90 INPUT"CODIFICAR(C) OU DECODI
FICAR(D):";E$
100 FOR L=1 TO 1000:NEXT
110 IF E$="C" THEN X=M
120 IF E$="D" THEN X=N
130 DIM D$(X)
140 FOR I=1 TO X
150 IF E$="C" THEN D$(I)=" "
160 FOR J=1 TO LEN(M$) STEP X
170 B$=MID$(M$,I+J-1,1)
180 D$(I)=D$(I)+B$
190 NEXT J
200 PRINT D$(I);:IF E$="C" THEN
 PRINT
210 NEXT I
220 END
```





Para usar o sistema de livro-código, tanto o emissor quanto o receptor da mensagem precisam ter uma cópia idêntica do dicionário fixo.

LIVRO-CÓDIGO

Até agora, trabalhamos apenas com cifras. Os códigos propriamente ditos consistem na substituição de uma palavra ou frase por outras. Isso exige que se recorra sempre a um livro-código, que deve ser guardado com muito cuidado, de modo a evitar que caia em mãos estranhas. Por esse motivo, geralmente, os textos são codificados e recebidos em lugares fixos, bem seguros.

O programa seguinte monta um dicionário de códigos de vinte palavras. Usando uma matriz bidimensional **A$(I,J)**, onde I = 1 guarda a palavra e I = 2 guarda seu código, primeiramente são lidas as informações após os comandos **DATA**. A parte seguinte, que vai da linha 120 à linha 170, recebe a palavra e imprime o código correspondente, ou faz o contrário, caso você tenha optado pela decodificação.

A mensagem FUGIR DE ROMA À MEIA-NOITE DE DOMINGO CHEGAR A NOVA YORK AO MEIO-DIA seria codificada como: 40476 10996 23874 12128 26569 10996 40553 68719 12128 54982 69783 11072. O texto codificado: 74891 22317 69783 74832 seria traduzido para ENVIAR DINHEIRO AO ANOITECER.

Com o próximo programa, você poderá codificar e decodificar rapidamente suas mensagens. Nas situações reais, onde são transmitidos textos muito grandes e os dicionários são volumosos, o computador exerce um papel fundamental, poupando bastante tempo.

```
 20 BORDER 0: PAPER 0: INK 7:
CLS
 25 POKE 23658,8
 30 PRINT TAB (10);"LIVRO DE C
ODIGOS"
 40 PRINT : PRINT : PRINT
 50 DIM a$(2,20,10)
 60 FOR i=1 TO 2
 70 FOR j=1 TO 20
 80 READ a$(i,j)
 90 NEXT j: NEXT i
100 INPUT "CODIFICAR (0) OU DE
CODIFICAR (1)";x
110 CLS
120 INPUT "Digite a palavra";m
$
125 IF LEN m$>=10 THEN  GOTO
130
127 FOR n=1 TO 10-LEN m$: LET
m$=m$+" ": NEXT n
130 IF m$="*" THEN  GOTO 280
140 FOR t=1 TO 20
150 IF m$=a$(1+x,t) THEN
PRINT a$(2-x,t)
160 NEXT t
170 GOTO 120
180 DATA "BRASILIA","LONDRES",
"PARIS","ROMA"
190 DATA "CHEGADA","SAIDA","VA
 PARA","FUJA PARA","SABADO"
200 DATA "DOMINGO","MEIO-DIA",
"AMANHECER","MEIA-NOITE"
210 DATA "ANOITECER","EM","AO"
```

Usando o código de multiplicação com diversas chaves, você poderá criptografar a mesma mensagem de várias maneiras. O texto cifrado é sempre apresentado em grupos de letras de comprimento fixo.

```
,"NO","ENVIE"
220 DATA "DINHEIRO","COMIDA"
230 DATA "54982","73581","9007
5","23874"
240 DATA "68719","68677","1032
7","40476"
250 DATA "27921","48553","1107
2","70355"
260 DATA "26569","74832","1099
6","12128"
270 DATA "69783","74891","2231
7","98724"
280 STOP
```

```
20 CLS
30 PRINT @8,"LIVRO DE CODIGO"
40 PRINT:PRINT:PRINT
50 DIM A$(2,20)
60 FOR I=1 TO 2
70 FOR J=1 TO 20
80 READ A$(I,J)
90 NEXT J,I
100 INPUT"CODIFICAR (0) OU DECO
DIFICAR (1)";X
110 CLS
120 INPUT"DIGITE A PALAVRA ";M$
130 IF M$="*" THEN END
140 FOR T=1 TO 20
150 IF M$=A$(1+X,T) THEN PRINT
A$(2-X,T)
160 NEXT
170 GOTO 120
180 DATA BRASILIA,LONDRES,PARIS
,ROMA
190 DATA CHEGADA,PARTIDA DE,VA
PARA,FUJA PARA,SABADO
200 DATA DOMINGO,MEIO-DIA,AMANH
ECER,MEIA-NOITE
210 DATA ANOITECER,EM,AO,NO,ENV
IE
220 DATA DINHEIRO,COMIDA
230 DATA 54982,73581,90075,2387
4
240 DATA 68719,68677,10327,4047
6
250 DATA 27921,48553,11072,7035
5
260 DATA 26569,74832,10996,1212
8
270 DATA 69783,74891,22317,9872
4
```

```
20  HOME
30  PRINT  TAB( 14)"LIVRO CODIG
O"
40  PRINT : PRINT : PRINT
50  DIM A$(2,20)
60  FOR I = 1 TO 2
70  FOR J = 1 TO 20
80  READ A$(I,J)
90  NEXT J
95  NEXT I
100  INPUT "CODIFICAR(0) OU DEC
ODIFICAR(1) ?";X
110  HOME
120  INPUT "DIGITE A PALAVRA ";
M$
130  IF M$ = "*" THEN  END
140  FOR T = 1 TO 20
150  IF M$ = A$(1 + X,T) THEN
PRINT A$(2 - X,T)
160  NEXT T
170  GOTO 120
180  DATA   NOVAYORK,LONDRES,PA
RIS,ROMA
190  DATA  CHEGAR,PARTIR,IR,FUG
IR,SABADO
200  DATA   DOMINGO,MEIO-DIA,TA
RDE,MEIA-NOITE
210  DATA    ANOITECER,DE,A,AO,
ENVIAR
220  DATA   DINHEIRO,SUPRIMENTO

230  DATA  54982,73581,90075,23
874
240  DATA  68719,68677,10327,40
476
250  DATA  27921,48553,11072,70
355
260  DATA  26569,74832,10996,12
128
270  DATA  69783,74891,22317,98
724
```

```
20 CLS
30 PRINT TAB(14)"LIVRO CÓDIGO"
40 PRINT:PRINT:PRINT
50 DIM A$(2,20)
60 FOR I=1 TO 2
70 FOR J=1 TO 20
80 READ A$(I,J)
90 NEXT J
95 NEXT I
100 INPUT "CODIFICAR(0) OU DECO
DIFICAR(1) ";X
110 CLS
120 INPUT "DIGITE A PALAVRA ";M
$
130 IF M$="*" THEN END
140 FOR T=1 TO 20
150 IF M$=A$(1+X,T) THEN PRINT
A$(2-X,T)
160 NEXT T
170 GOTO 120
180 DATA NOVAYORK,LONDRES,PARIS
,ROMA
190 DATA CHEGAR,PARTIR,IR,FUGIR
,SABADO
200 DATA DOMINGO,MEIO-DIA,TARDE
,MEIA-NOITE
210 DATA ANOITECER,DE,A,AO,ENVI
AR
220 DATA DINHEIRO,SUPRIMENTO
230 DATA 54982,73581,90075,2387
4
240 DATA 68719,68677,10327,4047
6
250 DATA 27921,48553,11072,7035
5
260 DATA 26569,74832,10996,1212
8
270 DATA 69783,74891,22317,9872
4
```

Apesar de estar limitado a apenas vinte palavras, o programa pode ser facilmente ampliado. Por exemplo, se você quiser fazer um dicionário de cinqüenta palavras, bastará trocar 20 por 50 nas linhas 50, 70 e 140. É claro que você terá também que fazer novas linhas **DATA**, com as palavras e códigos suplementares.

No programa atual para o Apple, você não poderá introduzir palavras com mais de dez caracteres. Caso queira ampliar este número para doze, por exemplo, você deve substituir 10 por 12 na linha 50. Essa restrição não existe nos outros programas.

PÁGINAS GRÁFICAS

Utilizando páginas gráficas, você poderá criar as mais diversas figuras e cenas animadas no microcomputador. Elas tornarão seus jogos muito mais divertidos.

Todos os tipos de animação relacionam-se a um fenômeno da percepção conhecido como persistência da visão. Cada imagem transmitida ao cérebro permanece "gravada" na memória por alguns instantes, mesmo que nossa visão já esteja captando uma nova imagem. Porém, quando uma série de imagens é mostrada com muita rapidez, o cérebro não consegue separá-las, pois não processa mais de doze imagens por segundo. As imagens parecem, então, sair umas das outras, e é exatamente isto que dá a impressão de movimento.

Certos livros infantis, cujas ilustrações compõem uma seqüência de imagens, permitem que se perceba esse fenômeno com clareza. Folheando-os rapidamente, suas figuras parecem se movimentar. Também o jogo de sombras chinesas, um dos mais remotos precursores do cinema, vale-se desse fenômeno: figuras recortadas ou criadas com as mãos são projetadas sobre paredes ou telas de linho em ritmo acelerado, formando uma sombra animada.

Finalmente, apesar dos progressos tecnológicos, o próprio cinema recorre ao mesmo princípio básico: os quadros, fixados em um filme, são projetados à velocidade de 25 unidades por segundo. Criam-se, assim, duas ilusões ópticas: a de que há uma corrente contínua de imagens, quando, na verdade, elas se sucedem de modo descontínuo; e a de que coisas imóveis têm movimento.

A montagem de um desenho animado é bem mais trabalhosa. O artista desenha sobre uma folha transparente. Ao mudar de quadro, sobrepõe a essa folha uma outra, e copia o desenho anterior, mudando ligeiramente sua forma. Tente calcular quantas cenas ele precisaria desenhar para produzir um desenho animado de uma hora de duração...

GRÁFICOS NO COMPUTADOR

Por que não usar o computador para agilizar esse processo, se até os micros mais simples são capazes de fazer desenhos de boa qualidade?

A capacidade gráfica dos micros atuais chega a ser impressionante. Entretanto, isso não é suficiente para satisfazer o público de um cinema. O custo da produção de um filme de ficção científica em computador chega a ser da ordem de milhões de dólares, devido aos equipamentos caríssimos que são usados. Esse preço só é compensador quando o roteiro requer cenas impossíveis de serem obtidas na vida real.

A grande maioria das pessoas não tem acesso a computadores profissionais de animação gráfica. Portanto, devem contentar-se em usar seus próprios micros para executar tais tarefas.

A animação gráfica é uma das muitas aplicações dos programas de projeto assistido por computador (PAC), já discutido em artigos anteriores. Nesses casos, o grau de sofisticação alcançado é limitado pela capacidade do microcomputador utilizado. Um dos mais poderosos equipamentos de animação gráfica para filmes, por exemplo, é o Cray X-MP, um supercomputador que opera à velocidade de 100 megaflops (100 milhões de operações em ponto flutuante por segundo). Mas, como as imagens são muito complexas e têm que ser trocadas muitas vezes por segundo, nem mesmo o Cray consegue gerar uma animação em tempo real. Suas imagens precisam ser filmadas separadamente, quadro a quadro, e recompostas, como se faz em um desenho animado.

Uma figura sem muitos detalhes, porém, permite animações bem próximas do real. Você mesmo já deve ter visto videogames bem produzidos, em que as imagens geradas chegam à velocidade de cinqüenta quadros por segundo.

O maior problema em uma animação feita no computador é a grande quantidade de informações existentes em um desenho. Quanto mais detalhada a figura, mais memória é exigida para seu armazenamento. Igualmente, quanto mais colorida for a imagem, mais espaço de memória RAM é necessário.

Devemos também levar em conta que, à medida que cresce a quantidade de informações a serem processadas pela UCP, mais lenta se torna a animação. É por esse motivo que nem mesmo os mais sofisticados computadores são auto-suficientes na produção de um filme ou desenho animado. Não existe ainda uma UCP tão rápida a ponto de dispensar, totalmente, o processo manual de montagem quadro a quadro.

Se a animação de figuras é uma tarefa lenta para os mais avançados computadores, como você poderá movimentá-las na tela de seu micro? Uma das soluções consiste em lançar mão das páginas gráficas.

- A PERSISTÊNCIA DA VISÃO
- A LANTERNA MÁGICA
- O DESENHO ANIMADO
- GRÁFICOS NO COMPUTADOR
- PÁGINAS GRÁFICAS
- MOVIMENTAÇÃO DE CUBO
- CRIANDO SUA PRÓPRIA ANIMAÇÃO

O QUE SÃO PÁGINAS GRÁFICAS

Todos os microcomputadores possuem uma área da memória reservada para a tela de vídeo. Ela pode ser de dois tipos: a *memória mapeada*, na qual a cada ponto da tela corresponde um local na memória; ou o *arquivo de códigos*, organizado como uma lista.

No conceito das páginas gráficas, em vez de se construir a figura diretamente na memória reservada para a tela, usamos uma outra área da memória RAM, definida especialmente para esse fim. Assim que se conclui a figura, ela é transferida para a parte da RAM normalmente associada à tela. A área utilizada para construir o desenho é chamada de *página gráfica*.

Mas, qual a vantagem de empregá-la? Seguramente não haverá nenhuma economia de tempo, se tivermos que desenhar a figura primeiro em uma área separada da memória. Ao contrário, quando se transferem as informações para a memória de tela, perde-se um certo tempo. A vantagem reside no fato de que podemos alterar à vontade a página "oculta", sem provocar simultaneamente alterações na tela.

É óbvio que a técnica de páginas gráficas não tem muito valor quando se trata de desenhar apenas uma figura. Suponhamos, porém, que você queira escrever um programa em que há um texto seguido de um desenho: seria bem mais conveniente ir construindo a figura em algum lugar da memória, enquanto o usuário estivesse ocupado com a leitura do vídeo. Com a tela gráfica, muito tempo seria economizado, uma vez que ela já estaria pronta e disponível em uma parte da memória.

Mas a grande vantagem das páginas gráficas se evidencia quando se pretende mostrar uma sucessão muito rápida de figuras. Os comandos em BASIC em geral escrevem apenas nas áreas reservadas para a tela. Isso significa que iremos construir a figura nesta área, e, depois, transportá-la para outra parte da RAM. Esse processo será mais lento na etapa de montagem, mas muito mais rápido na exibição da imagem, pois o microprocessador não precisará executar uma série de comandos e funções em BASIC. Ele apenas transferirá as informações daquela parte da memória para a tela. E, se você montar várias figuras em diferentes áreas da memória, elas poderão ser chamadas rapidamente para a tela, produzindo um efeito de animação.

O ALGORITMO DE UM CUBO

Tomemos como exemplo a montagem de uma seqüência animada que represente a rotação de um cubo. Decidiu-se que quatro quadros serão suficientes

para simular uma rotação e que o cubo dará cinco voltas.

O programa poderia ser estruturado da seguinte maneira:

```
para c = 1 até 5 faça
começo
limpar a tela
construir a figura número 1
limpar a tela
construir a figura número 2
limpar a tela
construir a figura número 3
limpar a tela
construir a figura número 4
fim
```

A idéia parece simples demais: limpa-se a tela, exibe-se cada uma das figuras em seqüência e o processo é repetido até que se completem as cinco rotações. Mas esse método apresenta uma desvantagem: os cálculos para se desenhar cada figura são refeitos a cada uma das cinco repetições. Como os cálculos tomam a maior parte do tempo no processo, ocorrerá um "pulo" entre cada figura exibida, o que resultará em um fraco efeito de animação.

Observe agora este algoritmo, que ilustra o procedimento geral usado na técnica de páginas gráficas:

```
limpar a tela
construir a figura número 1
guardar a tela na página de memória 1
limpar a tela
construir a figura número 2
guardar a tela na página de memória 2
limpar a tela
construir a figura número 3
guardar a tela na página de memória 3
limpar a tela
construir a figura número 4
guardar a tela na página de memória 4
para c = 1 até 5 faça
copie os dados da página 1 para a tela
copie os dados da página 2 para a tela
copie os dados da página 3 para a tela
copie os dados da página 4 para a tela
fim
```

O programa é mais longo e exige um demorado processo de construção das quatro figuras, antes de começar a animação. Mas, uma vez armazenadas nas páginas de memória, as figuras podem ser mostradas em rápida seqüência.

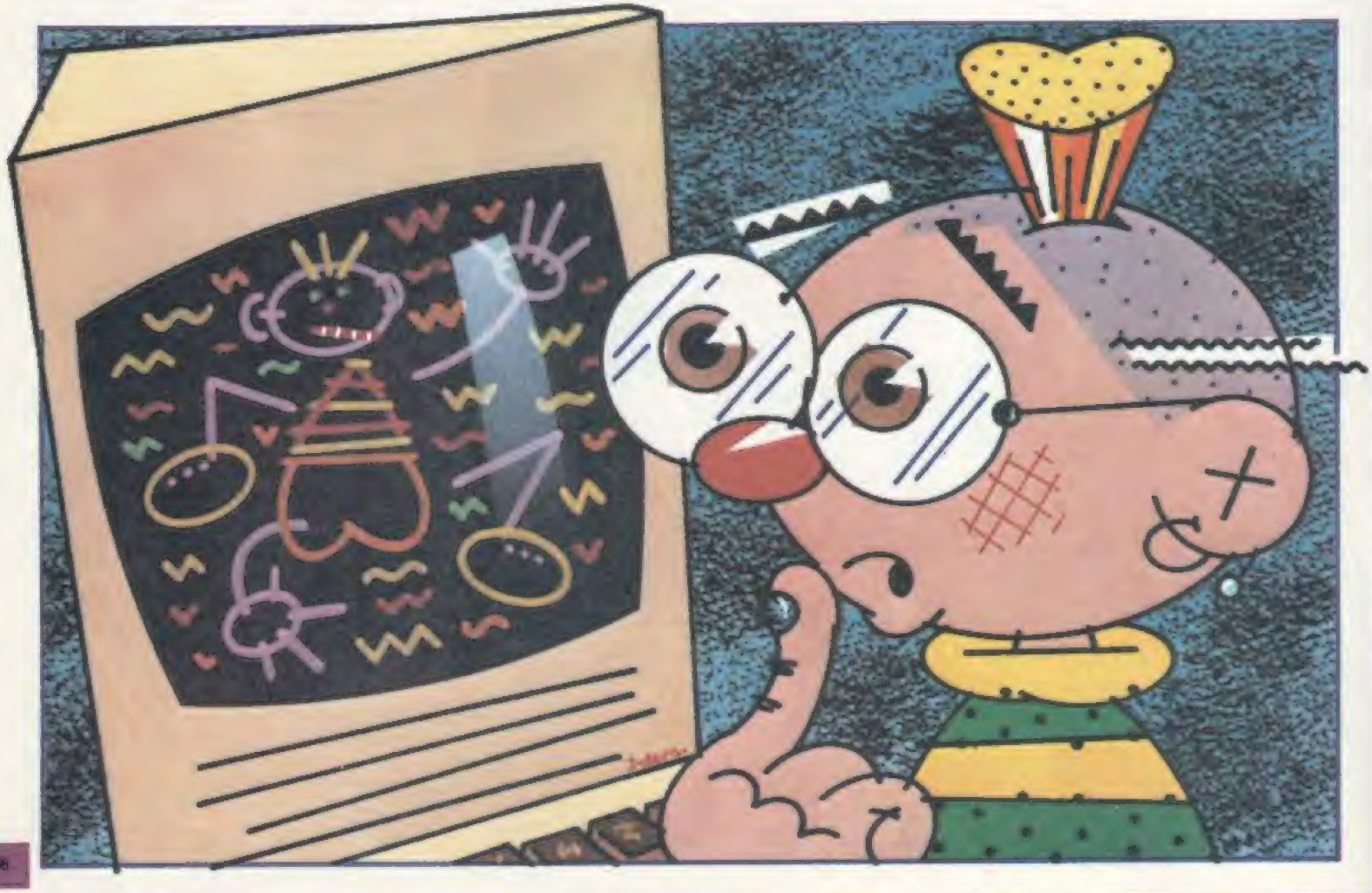
Embora a construção dos desenhos seja executada em BASIC, podemos utilizar uma rotina em código de máquina para efetuar a transferência de uma figura da tela para a página gráfica e vice-versa. Ela realizará essas transferências em um piscar de olhos — o que é a essência da animação.

FAÇA SUA PRÓPRIA ANIMAÇÃO

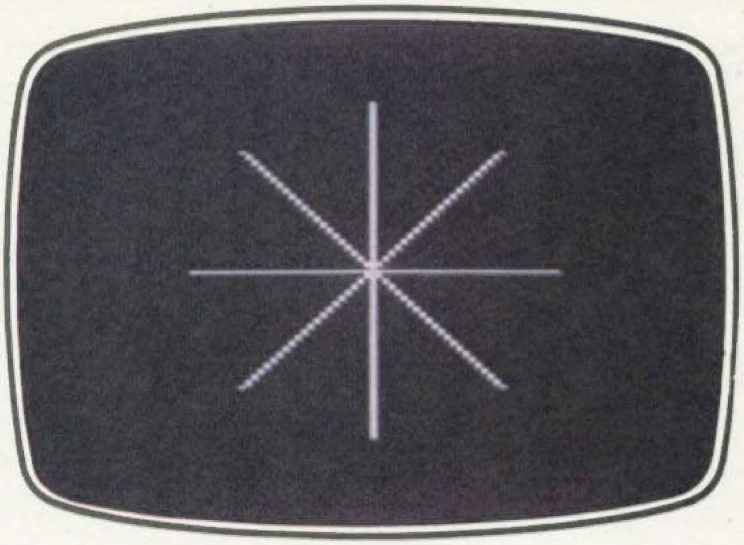
Os próximos programas são aplicações bem simples da técnica das páginas gráficas. Os que utilizam código de máquina devem ser gravados antes da execução, para evitar que se percam, caso ocorra algum erro de digitação. Você pode usar suas próprias figuras nesses programas, colocando os comandos de desenho nas linhas adequadas.

Em um próximo artigo, examinaremos em detalhe as técnicas aqui empregadas e você verá como aproveitar ao máximo a capacidade de seu micro.

Este programa — adequado só ao Spectrum de 48 K — produzirá a animação de um acrobata pulando sobre um



O acrobata na tela do Spectrum.



TRS-Color: o asterisco que se movimenta.

trampolim. Embora a maior parte do programa seja em BASIC, há um trecho em linguagem de máquina que dará a velocidade necessária às figuras na tela.

```
 10 BORDER 0: PAPER 0: INK 7:
 CLS
 20 CLEAR 53230
 30 GOSUB 220
 40 LET srce=64: LET dest=208
 50 CLS
 60 CIRCLE 128,168,7: PLOT 128
 ,161: DRAW 0,-15: DRAW -10,
 -10: PLOT 128,146: DRAW 10,
 -10: PLOT 118,161: DRAW 11,-5
 : DRAW 10,5
 70 PLOT 108,106: DRAW 40,0:
 PLOT 113,106: DRAW -8,-8:
 PLOT 145,106: DRAW 8,-8
 80 GOSUB 270: LET dest=dest+
  16
 90 PRINT AT 21,2;"qualquer te
 cla para comecar": PAUSE 0
100 CLS : CIRCLE 128,151,7:
PLOT 128,134: DRAW 0,-15: DRAW
-5,-16: PLOT 128,120: DRAW 5,
-17: PLOT 118,125: DRAW 10,5:
DRAW 11,-5
110 PLOT 108,106: DRAW 15,-4:
DRAW 10,0: DRAW 15,4: PLOT 113
,105: DRAW -8,-8: PLOT 144,105
: DRAW 8,-8
120 PRINT AT 6,4;"!!BOING!!"
130 GOSUB 270
140 PRINT AT 21,2;"qualquer te
cla para comecar": PAUSE 0
150 LET srce=208: LET dest=64
160 PRINT AT 17,3;"qualquer te
cla para pular": PAUSE 0
170 FOR n=0 TO 1
180 CLS
190 GOSUB 270: LET srce=srce+
 16
200 NEXT n
210 GOTO 150
220 DATA 1,0,16,17,0,0,33,0,0,
237,176,201
230 FOR i=53231 TO 53231+11
240 READ byte: POKE i,byte
250 NEXT i
260 RETURN
270 POKE 53236,dest
280 POKE 53239,srce
290 RAND USR 53231
300 RETURN
```

A linha 10 seleciona as cores da tela, da borda e do desenho: negro, negro e branco, respectivamente. A 20 reserva um espaço na memória e a 30 desvia o programa para a sub-rotina entre as linhas 220 e 260, que irá montar a rotina em linguagem de máquina, nessa área. Essa sub-rotina em BASIC lê os códigos após o comando **DATA** (linha 220) e os coloca na área reservada na memória, através do comando **POKE**.

A linha 40 define duas variáveis: **srce** e **dest**. A variável **srce** corresponde ao byte alto do endereço de onde os códigos devem ser retirados e **dest**, ao byte alto do endereço onde eles vão ser guardados. Informa-se, assim, ao computador onde ler a imagem da tela e onde colocá-la na memória. Em seguida, as linhas 60 e 70 desenham a primeira das duas figuras — o acrobata no ar. A linha 80 desvia o programa para a linha 270, onde há uma rotina que coloca os valores de **dest** e **srce** no programa em código de máquina. Depois, chama essa rotina para copiar a parte da tela em que está o acrobata.

O próximo passo consiste na criação da imagem para a segunda página de memória (linhas 100 a 120). Essa imagem é armazenada pela linha 130, que desvia o programa para a linha 270. A linha 150 troca os valores de **srce** e **dest**, o que faz com que os códigos sejam transferidos da RAM para a tela. As linhas 150 a 210 formam um laço que se encarrega de alternar as duas figuras na tela. Para interromper o programa, basta acionar **<BREAK>**.

Para montar sua própria animação (de dois quadros), você precisará modificar os comandos gráficos nas linhas 60 e 70 e 100 e 110. Futuramente, você verá que é possível usar até oito páginas gráficas em seqüência.

TRS-Color

O programa do TRS-Color é ligeiramente diferente dos destinados aos outros computadores. Ele usa três páginas gráficas — em vez de duas — das oito possíveis acessadas pelo comando **PCOPY**. Esse comando é utilizado da seguinte maneira: **PCOPY** *número da primeira página* **TO** *número da última página* — **PCOPY 1 TO 8**, por exemplo. Na verdade, a alternância de páginas é tão rápida, que se torna necessário introduzir uma pausa entre elas.

O programa mostra uma grande estrela girando continuamente.

```
10 PCLEAR 8:PMODE 2,1
20 SCREEN 1,1:CLS
30 C=ATN(1)/45
50 FOR N=0 TO 2
60 PCLS
70 FOR K=0 TO 360 STEP 45
80 LINE(127,95)-(127+59*SIN(C*(
K+N*15)),95-59*COS(C*(K+N*15)))
,PSET
90 NEXT
100 PCOPY 1 TO 3+N*2:PCOPY 2 TO
 4+N*2
110 NEXT
140 FOR N=3 TO 7 STEP 2
150 PCOPY N TO 1:PCOPY N+1 TO 2
160 FOR G=1 TO 30:NEXT G,N
170 GOTO 140
```

A linha 10 abre espaço para as oito páginas gráficas e seleciona **PMODE2** na página 1, para ficar em branco e preto com média resolução. Neste modo,

uma tela ocupa duas das páginas gráficas internas. Na linha 20 o modo de alta resolução gráfica é acionado.

As imagens que irão ocupar as três páginas são montadas da linha 30 até a linha 110, enquanto a linha 100 copia cada uma delas nas páginas internas. Uma tela ocupa duas páginas internas e as duas primeiras páginas são usadas para desenhar os gráficos.

As linhas 140 a 160 copiam as páginas armazenadas na tela, em seqüência. Há uma pausa na linha 160 para evitar que as imagens se alternem muito rapidamente e se sobreponham.

No Apple, contamos com duas páginas gráficas. O comando **HGR** ativa a alta resolução, limpa e mostra a página 1 da memória. O comando **HGR2** ativa a alta resolução, limpa e mostra a página 2. Colocaremos um desenho em cada página, fazendo com que se alternem. Digite e execute o programa. Você verá um beija-flor pairando no ar.

```
5  HGR
10  POKE  - 16302,0
20 X1 = 112:Y1 = 67:N = 16
30  GOSUB 100
40  FOR T = 1 TO 1000: NEXT T
50  HGR2
60 X1 = 115:Y1 = 83:N = 19
70  GOSUB 100
80  FOR T = 1 TO 1000: NEXT T
90  GOTO 300
100  FOR I = 1 TO N
110  READ X2
120  READ Y2
130  HPLOT X1,Y1 TO X2,Y2
140  LET X1 = X2: LET Y1 = Y2
150  NEXT I
160  RETURN
200  DATA      86,3,79,67,90,86,
192,3,176,64,141,99,198,166,170
,186
210  DATA  115,186,118,138,86,1
18,12,191,48,139,51,109,67,90,9
0,86
220  DATA  96,64,54,48,83,86,11
5,83,160,80,240,102,185,115
230  DATA  144,113,160,144,192,
147,179,169,144,185,115,148
240  DATA  86,118,12,191,48,139
,51,109,67,90,83,86
300  POKE  - 16304,0: POKE  - 1
6300,0
310  FOR I = 1 TO 70: NEXT I
320  POKE  - 16304,0: POKE  - 1
6299,0
330  FOR T = 1 TO 70: NEXT T
340  GOTO 300
```

A linha 5 ativa o comando **HGR**. Todos os comandos gráficos escreverão na primeira página. A linha 10 "fecha" a janela de quatro linhas na base da tela. A linha 20 define os pontos iniciais do primeiro desenho (**X1** e **Y1**) e o número de coordenadas a serem lidas (**N**). A linha 30 desvia o programa para a subrotina de desenho que vai da linha 100 à linha 160. Para construir esse primeiro desenho, serão lidas as linhas 200 e 210.

A linha 50 ativa o **HGR2**, abrindo espaço para se escrever na segunda página. A linha 60 define os valores iniciais da segunda figura e a 70 chama a subrotina de desenho, que agora irá ler as linhas 220, 230 e 240.

A rotina entre as linhas 300 e 340 alterna as duas figuras na tela. A linha 300 mostra a página 1 e a 320, a página 2. As linhas 310 e 330 introduzem pausas para diminuir a velocidade de alternância das imagens.

O programa do TK-2000 é semelhante ao do Apple. Devem ser feitas as seguintes modificacões:

```
5 MA: HOME :MP: HOME
10 MA: HGR2
50 MP: HGR2
300 MA
320 MP
```

A linha 5 limpa a primeira (**MA**) e a segunda (**MP**) páginas. A linha 10 habilita o usuário a escrever na primeira página, e a linha 50, na segunda. As linhas 300 e 320 mostram a primeira e a segunda páginas, respectivamente.

Para entender melhor o programa, veja as explicações para o Apple.

O MSX, como você já deve saber, possui uma memória exclusiva para a tela, a **VRAM**, que é dividida em oito partes de 2048 bytes. O comando **BASE** endereça cada arquivo de códigos em um deles, não necessariamente em seqüência. Como o modo de baixa resolução gráfica não exige muitas informações, grande parte da **VRAM** fica desocupada. Nesses espaços é possível criar até seis páginas gráficas. Para demonstrar isso, o próximo programa irá simular a rotação de um cubo.

```
5 BASE(18)=BASE(19)
10 BASE(17)=0:SCREEN 3
20 X1=127:Y1=40:N=4
30 GOSUB 200
40 BASE(17)=4096:SCREEN 3
50 X1=103:Y1=151:N=10
60 GOSUB 200
70 BASE(17)=6144:SCREEN 3
80 X1=95:Y1=151:N=10
90 GOSUB 200
100 BASE(17)=8192:SCREEN 3
110 X1=85:Y1=96:N=6
120 GOSUB 200
130 BASE(17)=10240:SCREEN 3
140 X1=155:Y1=151:N=10
150 GOSUB 200
160 BASE(17)=12288:SCREEN 3
170 X1=151:Y1=151:N=10
180 GOSUB 200
190 GOTO 400
200 FOR I=1 TO N
210 READ X2
220 READ Y2
230 LINE(X1,Y1)-(X2,Y2)
240 LET X1=X2:LET Y1=Y2
250 NEXT I
260 RETURN
300 DATA 177,96,127,151,74,96,1
27,40
310 DATA 133,96,176,96,151,151,
103,151,78,96,103,40,151,40,181
,96,128,96,103,40
320 DATA 110,96,170,96,155,151,
95,151,80,96,95,40,155,40,170,9
6,110,96,95,40
330 DATA 85,41,169,41,169,151,8
5,151,85,96,169,96
340 DATA 140,96,80,96,95,151,15
5,151,170,96,155,40,95,40,80,96
,140,96,155,40
350 DATA 121,96,73,96,103,151,1
51,151,181,96,151,40,103,40,73,
96,121,96,151,40
400 BASE(17)=0
410 FOR I=1 TO 70:NEXT I
420 BASE(17)=4096
430 FOR I=1 TO 70:NEXT I
440 BASE(17)=6144
450 FOR I=1 TO 70:NEXT I
460 BASE(17)=8192
470 FOR I=1 TO 70:NEXT I
480 BASE(17)=10240
490 FOR I=1 TO 70:NEXT I
500 BASE(17)=12288
510 FOR I=1 TO 70:NEXT I
520 GOTO 400
```

Primeiro, a linha 5 coloca o endereço de um arquivo que não iremos usar no último segmento da **VRAM**, junto com um outro, a fim de criar espaço. A linha 10 faz o computador colocar a tabela de padrões (a figura propriamente dita) a partir do endereço 0. A 20 define os pontos iniciais do desenho (**X1** e **Y1**) e o número de coordenadas que serão lidas (**N**). A 30 chama a rotina de desenho, que começa na linha 200 e termina na 260. A 300 contém as informações para o primeiro desenho.

Feitos os desenhos, o programa é desviado para a rotina da linha 400, que alterna as figuras na tela, dizendo ao computador, por meio do comando **BASE**, onde buscá-las. Essa rotina inclui ainda uma pausa entre as figuras, para que elas não se sobreponham.

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR LINHA

LINHA	FABRICANTE	MODELO
Apple II +	Appletronica	Thor 2010
Apple II +	CCE	MC-4000 Exato
Apple II +	CPA	Absolutus
Apple II +	CPA	Polaris
Apple II +	Digitus	DGT-AP
Apple II +	Dismac	D-8100
Apple II +	ENIAC	ENIAC II
Apple II +	Franklin	Franklin
Apple II +	Houston	Houston AP
Apple II +	Magnex	DM II
Apple II +	Maxitronica	MX-2001
Apple II +	Maxitronica	MX-48
Apple II +	Maxitronica	MX-64
Apple II +	Maxitronica	Maxitronic I
Apple II +	Microcraft	Craf II Plus
Apple II +	Milmar	Apple II Plus
Apple II +	Milmar	Apple Master
Apple II +	Milmar	Apple Senior
Apple II +	Omega	MC-400
Apple II +	Polymax	Maxxi
Apple II +	Polymax	Poly Plus
Apple II +	Spectrum	Microengenho I
Apple II +	Spectrum	Spectrum ed
Apple II +	Suporte	Venus II
Apple II +	Sycomig	SIC I
Apple II +	Unitron	AP II
Apple II +	Victor do Brasil	Elppa II Plus
Apple II +	Victor do Brasil	Elppa Jr.
Apple IIe	Microcraft	Craft IIe
Apple IIe	Microdigital	TK-3000 IIe
Apple IIe	Spectrum	Microengenho II
MSX	Gradiente	Expert GPC-1
MSX	Sharp	Hotbit HB-8000
Sinclair Spectrum	Microdigital	TK-90X
Sinclair Spectrum	Timex	Timex 2000
Sinclair ZX-81	Apply	Apply 300
Sinclair ZX-81	Engebras	AS-1000
Sinclair ZX-81	Filcres	NEZ-8000
Sinclair ZX-81	Microdigital	TK-82C
Sinclair ZX-81	Microdigital	TK-83
Sinclair ZX-81	Microdigital	TK-85
Sinclair ZX-81	Prologica	CP-200
Sinclair ZX-81	Ritas	Ringo R-470
Sinclair ZX-81	Timex	Timex 1000
Sinclair ZX-81	Timex	Timex 1500
TRS-80 Mod. I	Dismac	D-8000
TRS-80 Mod. I	Dismac	D-8001/2
TRS-80 Mod. I	LNW	LNW-80
TRS-80 Mod. I	Video Genie	Video Genie I
TRS-80 Mod.III	Digitus	DGT-100
TRS-80 Mod.III	Digitus	DGT-1000
TRS-80 Mod.III	Kemitron	Naja 800
TRS-80 Mod.III	Prologica	CP-300
TRS-80 Mod.III	Prologica	CP-500
TRS-80 Mod.III	Sysdata	Sysdata III
TRS-80 Mod.III	Sysdata	Sysdata Jr.
TRS-80 Mod.IV	Multix	MX-Compacto
TRS-80 Mod.IV	Sysdata	Sysdata IV
TRS-Color	Codimex	CS-6508
TRS-Color	Dynacom	MX-1600
TRS-Color	LZ	Color 64
TRS-Color	Microdigital	TKS-800
TRS-Color	Prologica	CP-400

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR NOME

FABRICANTE	MODELO	PAÍS	LINHA
Appletronica	Thor 2010	Brasil	Apple II +
Apply	Apply 300	Brasil	Sinclair ZX-81
CCE	MC-4000 Exato	Brasil	Apple II +
CPA	Absolutus	Brasil	Apple II +
CPA	Polaris	Brasil	Apple II +
Codimex	CS-6508	Brasil	TRS-Color
Digitus	DGT-100	Brasil	TRS-80 Mod.III
Digitus	DGT-1000	Brasil	TRS-80 Mod.III
Digitus	DGT-AP	Brasil	Apple II +
Dismac	D-8000	Brasil	TRS-80 Mod. I
Dismac	D-8001/2	Brasil	TRS-80 Mod. I
Dismac	D-8100	Brasil	Apple II +
Dynacom	MX-1600	Brasil	TRS-Color
ENIAC	ENIAC II	Brasil	Apple II +
Engebras	AS-1000	Brasil	Sinclair ZX-81
Filcres	NEZ-8000	Brasil	Sinclair ZX-81
Franklin	Franklin	USA	Apple II +
Gradiente	Expert GPC1	Brasil	MSX
Houston	Houston AP	Brasil	Apple II +
Kemitron	Naja 800	Brasil	TRS-80 Mod.III
LNW	LNW-80	USA	TRS-80 Mod. I
LZ	Color 64	Brasil	TRS-Color
Magnex	DM II	Brasil	Apple II +
Maxitronica	MX-2001	Brasil	Apple II +
Maxitronica	MX-48	Brasil	Apple II +
Maxitronica	MX-64	Brasil	Apple II +
Maxitronica	Maxitronic I	Brasil	Apple II +
Microcraft	Craft II Plus	Brasil	Apple II +
Microcraft	Caft IIe	Brasil	Apple IIe
Microdigital	TK-3000 IIe	Brasil	Apple IIe
Microdigital	TK-82C	Brasil	Sinclair ZX-81
Microdigital	TK-83	Brasil	Sinclair ZX-81
Microdigital	TK-85	Brasil	Sinclair ZX-81
Microdigital	TK-90X	Brasil	Sinclair Spectrum
Microdigital	TKS-800	Brasil	TRS-Color
Milmar	Apple II Plus	Brasil	Apple II +
Milmar	Apple Master	Brasil	Apple II +
Milmar	Apple Senior	Brasil	Apple II +
Multix	MX-Compacto	Brasil	TRS-80 Mod.IV
Omega	MC-400	Brasil	Apple II +
Polymax	Maxxi	Brasil	Apple II +
Polymax	Poly Plus	Brasil	Apple II +
Prologica	CP-200	Brasil	Sinclair ZX-81
Prologica	CP-300	Brasil	TRS-80 Mod.III
Prologica	CP-400	Brasil	TRS-Color
Prologica	CP-500	Brasil	TRS-80 Mod.III
Ritas	Ringo R-470	Brasil	Sinclair ZX-81
Sharp	Hotbit HB-8000	Brasil	MSX
Spectrum	Microengenho I	Brasil	Apple II +
Spectrum	Microengenho II	Brasil	Apple IIe
Spectrum	Spectrum ed	Brasil	Apple II +
Suporte	Venus II	Brasil	Apple II +
Sycomig	SIC I	Brasil	Apple II +
Sysdata	Sysdata III	Brasil	TRS-80 Mod.III
Sysdata	Sysdata IV	Brasil	TRS-80 Mod.IV
Sysdata	Sysdata Jr.	Brasil	TRS-80 Mod.III
Timex	Timex 1000	USA	Sinclair ZX-81
Timex	Timex 1500	USA	Sinclair ZX-81
Timex	Timex 2000	USA	Sinclair Spectrum
Unitron	AP II	Brasil	Apple II +
Victor do Brasil	Elppa II Plus	Brasil	Apple II +
Victor do Brasil	Elppa Jr.	Brasil	Apple II +
Video Genie	Video Genie I	USA	TRS-80 Mod. I

UM LOGOTIPO PARA CADA MODELO DE COMPUTADOR

INPUT foi especialmente projetado para microcomputadores compatíveis com as sete principais linhas existentes no mercado.
Os blocos de textos e listagens de programas aplicados apenas a determinadas linhas de micros podem ser identificados por meio dos seguintes símbolos:

Sinclair ZX-81

TRS-80

TK-2000

MSX

Spectrum

TRS-Color

Apple II

Quando o emblema for seguido de uma faixa, então tanto o texto como os programas que se seguem passam a ser específicos para a linha indicada.

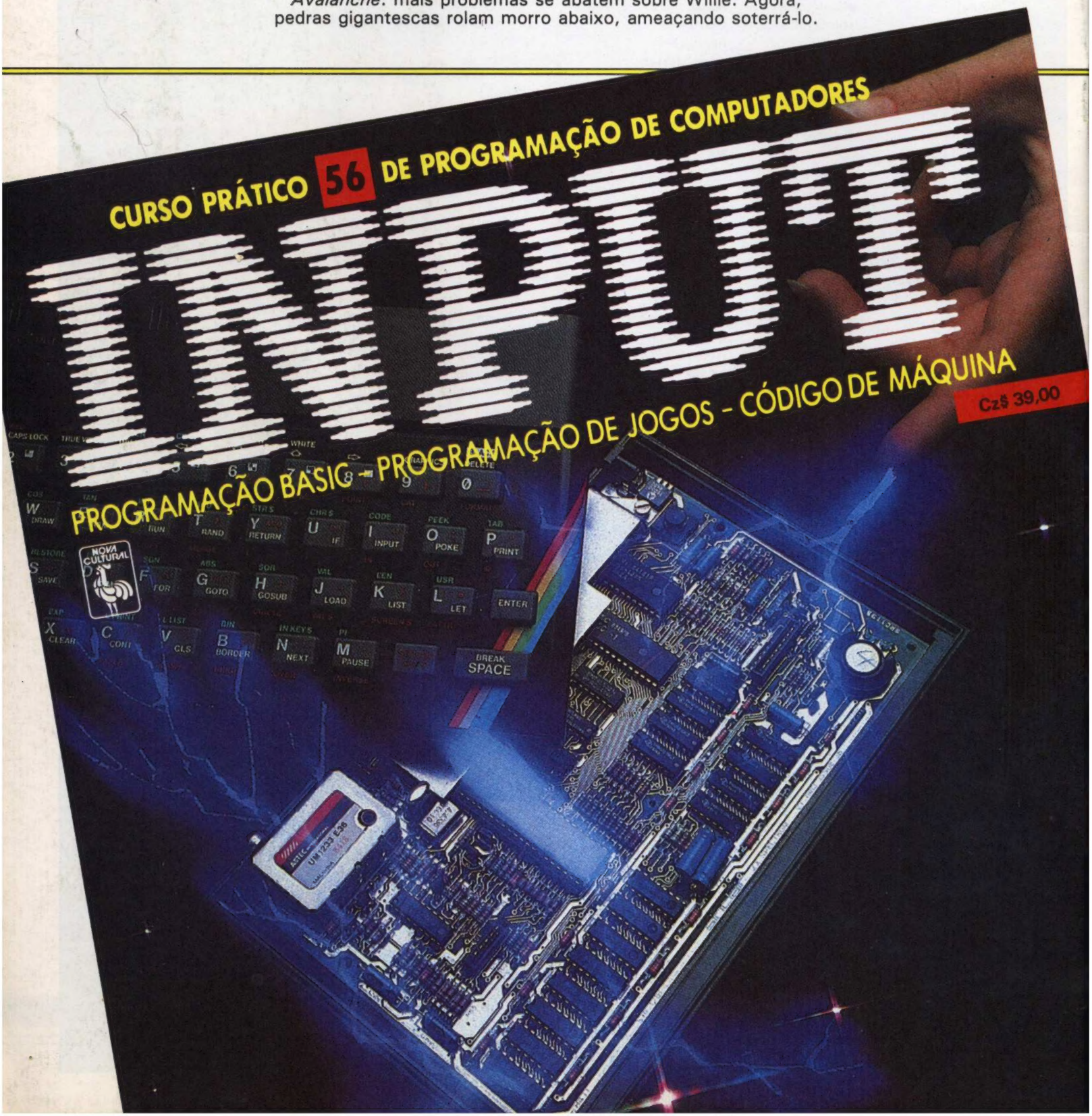
NO PRÓXIMO NÚMERO

PROGRAMAÇÃO BASIC

Veja como os programas são armazenados. Além de satisfazer sua curiosidade, você terá novos recursos para detectar erros.

APLICAÇÕES

Se você vive às voltas com números, prepare uma planilha de cálculo e peça socorro ao computador.

CÓDIGO DE MÁQUINA

Avalanche: mais problemas se abatem sobre Willie. Agora, pedras gigantescas rolam morro abaixo, ameaçando soterrá-lo.